SEMANTIC SPEECH PROCESSING WITH NEURAL NETWORKS

LASSE BORGHOLT

February 2022

This thesis has been submitted to the
PhD School of The Faculty of Science, University of Copenhagen

ACADEMIC SUPERVISION:

Christian Igel
Supervisor, University of Copenhagen

Anders Søgaard
Co-supervisor, University of Copenhagen

INDUSTRIAL SUPERVISION:

Lars Maaløe
Supervisor, Corti & Technical University of Denmark

Željko Agić
Co-supervisor, Unity Technologies

AFFILIATIONS:

University of Copenhagen & Corti

ASSESSMENT COMMITTEE:

Isabelle Augenstein
University of Copenhagen

Karen Livescu
Toyota Technological Institute at Chicago

Florian Metze
Carnegie Mellon University

Lasse Borgholt, PhD thesis:
Semantic Speech Processing with Neural Networks

ABSTRACT

Human spoken language understanding relies heavily on contextual information. The context of a single word provides important clues for the listener to accurately recognize and understand it. If a word is mispronounced or drowned out by noise, the listener may infer the word from context. Words like `park` and `play` have different meanings depending on the context they appear in. And even when a word is completely unknown to the listener, context may help in deriving its meaning. Thus, training models to identify semantic relations from context is an important path towards computers that can mimic the human understanding of spoken language.

This idea has a long tradition in neural representation learning. Here, the goal is to learn data representations that are useful for other machine learning tasks. For example, in text-based natural language processing, the idea has inspired approaches for learning semantic word embeddings, such as word2vec. And more recently, it has inspired the development of masked language models, such as BERT. These approaches have revolutionized natural language processing. During the course of this thesis project, speech processing has undergone a similar development. However, these models are still evolving and there is much we do not know about what they learn, why they work, and how we can improve them.

This thesis investigates machine learning models that learn semantic features directly from speech. The first part of the thesis studies supervised learning and shows the following.

(i) The performance of end-to-end speech recognition models depends heavily on access to contextual information.

(ii) Question tracking and symptom detection in spoken medical dialogues benefit from multimodal input.

The second part of the thesis focuses on unsupervised learning. The contributions are the following.

(iii) An overview of unsupervised representation learning for neural speech processing and a corresponding model taxonomy.

(iv) A comparison and analysis of two generations of the popular wav2vec framework for low-resource speech recognition.

(v) A novel hierarchical latent variable model which is benchmarked against other popular stochastic and deterministic models.

(vi) A comparison of contextualized speech representations and speech recognition transcripts when used as input for spoken language understanding tasks.

ABSTRACT IN DANISH - RESUMÉ PÅ DANSK

Menneskets sprogsforståelse er stærkt afhængig af kontekstuel information. Konteksten af et enkelt ord giver vigtige ledetråde for lytteren til nøjagtigt at genkende og forstå det. Hvis et ord udtales forkert eller overdøves af støj, kan lytteren udlede ordet ud fra konteksten. Ord som `frø` og `får` har forskellige betydninger afhængigt af den kontekst, de optræder i. Og selv når et ord er helt ukendt for lytteren, kan kontekst hjælpe med at udlede dets betydning. At træne modeller til at identificere semantiske relationer fra kontekst er således en vigtig vej mod computere, der kan efterligne den menneskelige forståelse af det talte sprog.

Denne idé har en lang tradition inden for repræsentationslæring med neurale netværk. Her er målet at lære data repræsentationer, der er nyttige til andre maskinlæringsopgaver. Indenfor tekstbaseret sprogteknologi har ideen inspireret modeller som word2vec, der lærer semantiske repræsentationer af ord. Og i nyere tid har den givet inspiration til udviklingen af såkaldte maskerede sprogmodeller, såsom BERT. Disse tilgange har revolutioneret sprogteknologien. I løbet af dette forskningsprojekt har talebehandling gennemgået en lignende udvikling. Disse modeller er dog stadig på et tidligt udviklingsstadie, og der er meget vi ikke ved om, hvad de lærer, hvorfor de virker, og hvordan vi kan forbedre dem.

Denne afhandling undersøger maskinlæringsmodeller, der lærer semantiske funktioner direkte fra tale. Første del af afhandlingen undersøger superviseret læring og viser følgende.

(i) Moderne talegenkendelsesmodellers evne til nøjagtigt at genkende ord, er i høj grad afhængig af kontekstuel information.

(ii) Maskinlæringsmodeller til spørgsmålssporing og symptomdetektion i nødopkald kan med fordel benytte multimodalt input.

Den anden del af afhandlingen fokuserer på usuperviseret læring. Bidragene er som følger.

(iii) En omfattende oversigt over usuperviseret repræsentationslæring til neural talebehandling samt en modeltaksonomi.

(iv) En sammenligning og analyse af modeller fra to generationer af wav2vec tilgangen til talegenkendelse med få ressourcer.

(v) En ny hierarkisk latent variabel model som sammenlignes med andre populære stokastiske og deterministiske modeller.

(vi) An sammenligning af kontekstualiserede talerepræsentationer og talegenkendelsestransskriptioner når brugt som input til sprogsforståelsesopgaver.

PUBLICATIONS

This thesis consists of the following papers. Each paper makes up a chapter of the thesis. Published papers have been reformatted to fit the layout, but the content has not been changed.[1] The following papers are included:

[i] **Lasse Borgholt**, Jakob D. Havtorn, Željko Agić, Anders Søgaard, Lars Maaløe, and Christian Igel. "Do End-to-End Speech Recognition Models Care About Context?" In: *Proceedings of the 21st Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020, pp. 4352–4356.

[ii] Jakob D. Havtorn, Jan Latko, Joakim Edin, **Lasse Borgholt**, Lars Maaløe, Lorenzo Belgrano, Nicolai Jacobsen, Regitze Sdun, and Željko Agić. "MultiQT: Multimodal Learning for Real-Time Question Tracking in Speech." In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2020, pp. 2370–2380.

[iii] **Lasse Borgholt**, Jakob D. Havtorn, Joakim Edin, Lars Maaløe, and Christian Igel. "An Overview of Unsupervised Neural Speech Representation Learning." Short version in: *The 2nd Workshop on Self-supervised Learning for Audio and Speech Processing (SAS) at AAAI*. 2022. Extended version under review.

[iv] **Lasse Borgholt**, Tycho M. S. Tax, Jakob D. Havtorn, Lars Maaløe, and Christian Igel. "On Scaling Contrastive Representations for Low-Resource Speech Recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 3885–3889.

[v] Jakob D. Havtorn, **Lasse Borgholt**, Søren Hauberg, Jes Frellesen, and Lars Maaløe. "Benchmarking Generative Latent Variable Models for Speech." *Under review*.

[vi] **Lasse Borgholt**, Jakob D. Havtorn, Mostafa Abdou, Joakim Edin, Lars Maaløe, Anders Søgaard, and Christian Igel. "Do We Still Need Automatic Speech Recognition for Spoken Language Understanding?". *In preparation*.

1 The reformatted versions are best read using a two-page layout.

ACKNOWLEDGMENTS

First of all, I want to thank *all* my supervisors. Christian, you have been an amazing support and help throughout this project. Your rigor is admirable and I consider your name a quality stamp on the work we did over the last three years. Lars, the level of energy you bring is otherworldly and completely indispensable. I look forward to many more constructive discussions in the coming years.

I also want to thank all the amazing people part of the CoAStaL group at the University of Copenhagen. My co-supervisor, Anders, deserves immense praise for fostering an inclusive, open, and relaxed culture. Anyone who aspires to bring together a great team could learn from this. Due to pandemics and whatnot, my presence at the university has sadly been too limited during the last two years.

I am grateful to be part of a company like Corti, which is capable of embracing and supporting people like me, who wants to learn and develop within their field. I want to thank Lars, Andreas, Michael, and Anders, who founded the company. I also want to thank the machine learning team members anno 2018, Tycho, Marco, and Jan, who inspired me to take on this challenge. Last but not least, I want to thank Jakob, with whom I had countless inspiring discussions during the last three years. It has been a blessing to have you as a fellow PhD student at Corti.

Finally, I want to thank my family, my friends, and Caroline who have provided indispensable support and much needed distraction throughout the years.

The project was funded by Corti and Innovation Fund Denmark through an industrial PhD grant (no. 8053-00184B).

CONTENTS

I BACKGROUND 1

1 INTRODUCTION 3
- 1.1 Motivation: Cardiac arrest detection 4
- 1.2 Challenges in conversational SLU 6
- 1.3 Scope of the thesis 8
- 1.4 Semantic speech processing 9

2 TECHNICAL BACKGROUND 13
- 2.1 Connectionist temporal classification 13
- 2.2 Wav2vec 2.0 17

3 MAIN CONTRIBUTIONS 21

II SUPERVISED LEARNING 23

4 DO END-TO-END SPEECH RECOGNITION MODELS CARE ABOUT CONTEXT? 25
- 4.1 Introduction 25
- 4.2 End-to-End Speech recognition 27
- 4.3 Method 30
- 4.4 Experiments 32
- 4.5 Conclusions 35

5 MULTIQT: MULTIMODAL LEARNING FOR REAL-TIME QUESTION TRACKING IN SPEECH 37
- 5.1 Introduction 37
- 5.2 Multimodal speech labeling 39
- 5.3 Data 42
- 5.4 Experiments 43
- 5.5 Discussion 48
- 5.6 Related work 51
- 5.7 Conclusions 52

III UNSUPERVISED LEARNING 53

6 AN OVERVIEW OF UNSUPERVISED NEURAL SPEECH REPRESENTATION LEARNING 55
- 6.1 Introduction 55
- 6.2 Unsupervised representation learning 56
- 6.3 Self-supervised models 57
- 6.4 Probabilistic latent variable models 67
- 6.5 Evaluation procedures 75
- 6.6 Discussion 81
- 6.7 Conclusion 84

7 ON SCALING CONTRASTIVE REPRESENTATIONS FOR LOW-RESOURCE SPEECH RECOGNITION 85
- 7.1 Introduction 85

7.2 Contrastive learning for speech 87
7.3 Bidirectional extension 88
7.4 Results 91
7.5 Conclusions 93
8 BENCHMARKING GENERATIVE LATENT VARIABLE MODELS FOR SPEECH 95
8.1 Introduction 95
8.2 Latent variable models for speech 96
8.3 Speech modeling benchmark 104
8.4 Phoneme recognition 107
8.5 Conclusion 112
9 DO WE STILL NEED AUTOMATIC SPEECH RECOGNITION FOR SPOKEN LANGUAGE UNDERSTANDING? 113
9.1 Introduction 113
9.2 Tasks 115
9.3 Experiments 117
9.4 Results 120
9.5 Discussion 123
9.6 Conclusion 123

IV DISCUSSION AND CONCLUSIONS 125
10 DISCUSSION 127
10.1 Chapter 4 revisited: The case for context 127
10.2 Chapter 5 revisited: Looking beyond transcripts 129
10.3 Chapter 7 revisited: Digging deeper in wav2vec 2.0 129
11 CONCLUSIONS AND OUTLOOK 133

V APPENDIX 137
A APPENDIX FOR CHAPTER 8 139
A.1 Reproducibility statement 139
A.2 Ethics statement 139
A.3 Datasets 140
A.4 Model architectures 140
A.5 Training details 142
A.6 Converting the likelihood to units of bits per frame 143
A.7 Additional likelihood results 144
A.8 Additional discussion on Gaussian likelihoods in LVMs 148
A.9 Additional discussion on the output distribution 149
A.10 Additional graphical models 149
A.11 Additional latent evaluation 150
A.12 Distribution of phoneme duration in TIMIT 152
A.13 Model samples and reconstructions 152

BIBLIOGRAPHY 157

Part I

BACKGROUND

1 INTRODUCTION

It has been a longstanding ambition to build machines that can understand spoken language. The first system for translating speech to meaningful symbols was built at Bell Labs in 1952 [133]. It was a single-speaker system capable of recognizing digits zero to nine. Today, automatic speech recognition is still an open area of research. Modern speech processing relies on high-performance parallel computing [40, 157], large volumes of data [136, 213], and years of innovation in model design [109, 266]. The first speech recognition system to outdo humans in a single domain was presented years ago [277], and speech recognition has only improved since then [11, 13, 94].

But high-quality computer-enabled speech recognition does not necessarily mean that machines now understand human language. For instance, a model capable of recognizing digits does not have to learn that 1 is greater than 0, 2 is greater than 1, and so on. The notion that one digit is larger than another relates to the meaning of the symbol rather than the symbol itself. The same reasoning extends beyond numbers. The representation of a word does not necessarily convey anything about its meaning. In natural language processing, it is common to represent words and other lexical units as one-hot vectors. See figure 1. With this representation, any two distinct words will be as different from each other, as any other pair of words, even if the words are completely interchangeable to humans. Thus, the one-hot representation conveys nothing but the identity of the word.

The research presented in this thesis concerns semantic speech processing. To label any speech processing task as semantic is not done without some ambiguity. And to claim that a machine learning model *understands* spoken language is even more problematic. In order to al-

CAT	[	0	0	0	0	0	0	**1**	0	0	0	0	0	0]
DOG	[	0	0	0	0	0	0	0	0	0	0	0	0	**1**]
CUT	[	0	0	**1**	0	0	0	0	0	0	0	0	0	0]

Figure 1: Three one-hot vectors each corresponding to a word. Even though `cat` and `dog` represent somewhat similar concepts, the distance between them is the same as between `cut` and any one of them. This will be true for any distance metric that does not make prior assumptions about the index of the non-zero entry. From an acoustic point-of-view, `cat` and `cut` will typically be much closer.

low for a fruitful discussion of this subject, a more careful treatment of this theme is required. First, however, we should built an understanding of the motivation behind the research project which was formulated in collaboration with the Danish machine learning start-up Corti. Thus, to do so, it is helpful to review the challenges faced by Corti prior to February 2019 (sec. 1.1 and 1.2), when this project started. These challenges are used to outline the scope of the project (sec. 1.3), before we return to discuss what should be understood by semantic speech processing (sec. 1.4).

1.1 MOTIVATION: CARDIAC ARREST DETECTION

In 2016, Corti started to work on a machine learning-based technology assisting 911 call takers in identifying out-of-hospital cardiac arrests during emergency calls. The system should be able to process a call in real-time and provide feedback to the call taker if a cardiac arrest is suspected. This task is not the subject of this thesis, but the challenges associated with building the machine learning models have inspired the work presented in the following chapters.

Cardiac arrest is a critical condition where fast recognition can significantly increase the chance of survival [248], but also a condition that is notoriously difficult to spot in certain cases [29]. The primary indicators of a cardiac arrest, when assessed over the phone, are found in the caller's description of the situation. Thus, a cardiac arrest detection model should be able to extract information from the interaction between call taker and bystander. What are they saying? What does it mean? Such a model is primarily intended to be helpful in cases that are not immediately obvious to a trained call taker. Consider a very simple example:

Q: Is he breathing?
A: Yes

This is typically one of the first questions to be asked, if the call taker suspects a cardiac arrest. In case of a clear-cut answer, as above, the call taker is unlikely to need assistance from a machine learning model. However, for such a model to remain credible, it is still important that simple question-answer pairs are correctly interpreted. If the answer is *no*, the model should trigger an alarm, and if the answer is *yes*, the model should do nothing.

In order to do this, the model needs to learn that *he* refers to the patient, that the absence of *breathing* is an important indicator of a cardiac arrest, that *is he breathing* is a question, and that *yes* is the answer to this question. Unlike in the written example above, no question marks will be available to the model through speech, and it will also have to identify who is saying what. Furthermore, there is no guar-

antee that the bystander is capable of answering yes or no. He or she might be stressed and use misleading filler words.

Q: Is he breathing?
A: Eh-em. Yes, let me check. One sec.

In this example, *yes* is not the answer to the question, but merely used to confirm that the bystander understands the question. Here, the call taker will also understand the answer just fine, but a simple machine learning model might not; it is not enough to learn, if the question *is he breathing* is followed by the word *yes*. At this point in the conversation, there will probably not be enough indicators to suspect a cardiac arrest, so the model should not trigger an alarm. Now, let us see an example where the model is more likely to be able to help the call taker.

Q: Is he breathing?
A: Yeah, but he is sort of gasping.

Here, the bystander gives a positive answer to the question, but adds an additional piece of information. Gasping might be a sign of *agonal breathing*; a reflex that can be triggered when the brain is not getting enough oxygen [57]. This description might lead the call taker to conclude that the patient is not in need of cardiopulmonary resuscitation (CPR) given the positive answer. A machine learning model, capable of reviewing many more historical calls than a human, might be able to spot this symptom and encourage the call taker to ask further clarifying questions. However, agonal breathing is a rare condition that most people will never have encountered. Another bystander might use words like *snoring* or *hissing* to describe the breathing of the patient. Thus, the model needs to learn that *gasping* is similar to these words in the given context.

The examples above illustrate the case for cardiac arrest detection. In many aspects, this problem is a classic spoken language understanding (SLU) task. SLU systems can generally be regarded as systems for extracting semantic information from speech [263]. At the time Corti started to work on cardiac arrest detection, most SLU systems consisted of a speech recognition model and a downstream model [181]. The speech recognition model converts input speech to a transcript, and the downstream model processes the transcripts in order to classify the input (e.g., cardiac arrest or not). Building a speech recognition model in this domain (i.e., emergency calls) was a major hurdle, as training data for the model was not readily available. Furthermore, even a state-of-the-art speech recognition system might not be the best prerequisite for solving the downstream task. These challenges have formed the basis for this thesis. In the paragraphs below, we discuss them in more detail.

1.2 CHALLENGES IN CONVERSATIONAL SLU

We first discuss problems related to obtaining parallel data for conversational speech recognition. We use the term *conversational* to cover any spontaneous interaction between two or more individuals, although it might sound a bit casual with regard to emergency calls. Then, we consider the implications of not having access to large text-only resources, which are commonly used to improve speech recognition through language modeling. Finally, we consider the limitations of speech recognition and text as a representation for downstream tasks.

1.2.1 *Transcribing conversations*

In order to obtain data for training a speech recognizer, thousands of working hours have to be invested in transcribing and segmenting speech. This process is costly and error prone. Furthermore, spontaneous speech in emergency calls are not easily put into text. There are at least two speakers involved and the conversation is typically very stressful. Speech may overlap, words may be fragmented or mispronounced, and noise may drown out the speech from time to time.

These problems are commonly addressed through rigid transcription guidelines, which might be difficult to model or lead to ambiguity. If a word is mispronounced, such that it sounds like a different one, which word do we want in our transcript? The one that corresponds to the acoustics or the intended word? For SLU tasks, the latter might be preferred, but at the same time, it is problematic if the model is trained to second-guess human intentions.

Other problems are similar to this one, when it comes to representing conversational speech. How do we handle background noise and unintelligible spoken noise? And what about overlapping speech? Of course, gold transcripts can be simplified to the point that modeling them is straightforward, but important details might get lost. Unsurprisingly, most research on speech recognition make use of "text first" data [213, 220]. That is, datasets created from text resources where the text is read aloud and recorded (e.g., audio books).

1.2.2 *Lack of text-only data*

Modern speech recognition models rely on language models trained on text-only data. This is also true for the so-called end-to-end models that will be investigated in this thesis. It is not uncommon to find that decoding the output of a speech recognizer with a language model can reduce the word error rate (WER) with more than 50% [99, 191].

Language model decoding is particularly beneficial when parallel data (i.e., speech utterances with gold transcripts) are limited. How-

ever, it requires access to large amounts of text-only data. Some of the most popular datasets for speech recognition research [213, 220] come with massive resources for training external language models. This text data are derived from the same source material as the parallel data used to train the speech recognizer. It is a considerable advantage, if training data for the speech recognition model and the language model come from the same domain [178].

For conversational speech recognition, this scenario can only be simulated. That is, text data are obtained by transcribing speech, and thus, in-domain conversational text-only data do not exist. Text might be obtained from other written sources, such as online chat fora, but the nature of the text will be very different. For emergency calls in particular, it is unlikely to find text-only data with the same characteristics. Many of the artifacts discussed above (e.g., background noise, overlaps and interruptions) simply do not apply. And conversely, writing contains artifacts that do not apply to speech (e.g., emojis and abbreviations). Thus, there will be a considerable domain mismatch.

1.2.3 *Meaning beyond text*

In the two previous paragraphs, we discussed problems related to building a speech recognition model for conversational speech. The primary reason we are concerned with speech recognition in the first place, is because we want to use the automatically generated transcript as input for a downstream task, such as cardiac arrest detection. As we will see in chapter 5, it is easier to extract semantic information from a transcript than from the raw audio. Text has a well defined structure where units that carry semantic information (i.e., words) are clearly separated, such that it is easy to process with a computer. Furthermore, text processing requires much less computation than speech processing, due to the sparsity and the relatively short sequence length of text data.

On the other hand, some downstream tasks benefit from information strictly related to the speech signal. For instance, speaker identity, emotion, and intonation are essentially discarded by a speech recognition model. Furthermore, as we discussed above, text on its own convey little about the meaning of the words that make up a sentence. For instance, if we consider the words `cat` and `dog` in isolation, there is no way of knowing that both represent an animal commonly kept as a pet. Neither through a basic one-hot representation, as illustrated in figure 1, or through the corresponding character representation. A downstream model trained on speech recognition transcripts will probably be able to learn this relation, if relevant. But such a model will only be able to learn from the labeled examples available for the downstream task. And there is no guarantee that this data contain

enough mentions of cats and dogs to learn this. Instead, we might be able learn such relations in an unsupervised manner.

Although the `cat-dog` example is a primitive simplification, the development of text-based natural language understanding (NLU) has largely been inspired by this ambition [68, 194]. The general approach is to train a deep neural network to predict masked parts of the input – a so-called masked language model. This model can then be fine-tuned for a downstream task. If we have enough unlabeled speech data, we could use this approach to train a masked language model on the speech recognition transcripts. However, this would still require enough parallel data to train a speech recognizer in the first place. And even if data are available, the speech recognition transcripts will contain a significant number of errors that will likely degrade the quality of the masked language model.

1.3 SCOPE OF THE THESIS

There are many ways to tackle the challenges described above, but it is not possible to explore them all. For instance, in order to the address the sparsity of transcribed conversations, one could explore the use of pseudo labels obtained from speech recognition models trained on limited data. This approach is known as self-training or pseudo-labeling, and has seen tremendous progress in recent years [135, 278]. The lack of text-only data could be addressed through domain adaptation for language models trained on out-of-domain data [178, 195]. And as already discussed, semantic representations could be learned by training text-based masked language models on the noisy speech recognition transcripts [159].

The focus of this thesis is the ability of deep neural networks to learn semantic features directly from speech. In the section that follows, the notion of semantic speech processing will be explicated. Both supervised (part ii) and unsupervised methods (part iii) will be studied in this thesis, although the latter is of particular interest. Unsupervised representation learning for speech has seen significant progress during the three years of this project [11, 13, 111]. In general, representation learning is concerned with learning features that can benefit downstream tasks, such as cardiac arrest detection. In contrast to the methods sketched out above, it has the potential to completely circumvent the need for parallel data. In addition, it has been able to realize the promise of high-quality low-resource speech recognition. Finally, speech representations offer a natural way to encode features that are not captured by text.

1.4 SEMANTIC SPEECH PROCESSING

The term *semantic speech processing* is used as an umbrella term to capture the work collected in the following chapters. It is not used explicitly in the individual studies (chapters 4-9), but in the concluding chapter, it will be used as the common denominator for discussing this thesis in its entirety. In this section, the term will be developed through short discussions of tasks that are generally assumed to require semantic language understanding (section 1.4.1), and the idea that machine learning models can learn semantic representations (section 1.4.2). Finally, based on this brief discussion, a working definition of semantic speech processing is proposed (section 1.4.3).

But first, it makes sense to look beyond machine learning and address a general idea of semantics. In particular, the contextual approach to lexical semantics captures what we strive to learn in semantic speech processing and other areas of machine learning. In this theory, the notion of context is particularly important: *"it is assumed that the semantic properties of a lexical item are fully reflected in appropriate aspects of the relations it contracts with actual and potential contexts"* ([63], p. 1). As we will see in the following chapters, models that learn from context have come to play a crucial role in modern speech processing, as they have been doing in text-based natural language processing for several years.

1.4.1 *Semantic tasks*

Some tasks aim to directly extract semantics. Above, SLU systems were defined as systems for extracting semantic information from speech [263]. Thus, such tasks are typically labeled SLU tasks. The most common examples are intent classification and slot filling. Consider the sentence "Switch on the bathroom lights" from the Fluent Speech Commands (FSC) dataset [181]. In slot filling, we want to extract values corresponding to pre-defined slots. For this example, we have the following slot-value pairs: `action: activate`, `object: lights`, and `location: washroom`. For intent classification, the slot values are combined to form a single category, such that the task can be phrased as a simple classification problem.

If a model can learn to identify the correct intent given different ways of phrasing it (e.g., "Turn the washroom lamp on"), it may seem reasonable to assume that the model has learned a semantic representation of the utterance. However, widely used datasets for these tasks, such as FSC [181] and ATIS [103], reuse the same sentences in train and test sets, such that only the speaker differs. As a result, these tasks amount to sentence recognition and only evaluate speaker generalizability [5]. Another such dataset, Audio SNIPS [62], relies on synthesized speech rather than natural speech. As a consequence, a

lot of recent effort has gone into organizing, developing, and rephrasing new and existing SLU tasks [5, 81, 246, 280].

Where SLU is used to cover tasks where the targets commonly constitute semantic categories, semantic speech processing apply more broadly. For instance, speech recognition is commonly not regarded an SLU task [246]. As discussed in the introduction, text does not convey much semantic information on its own. On the other hand, the segmentation of speech into words or characters does make it a lot easier to learn relations such as the one between the words `cat` and `dog`. Important speech recognition features can be learned just by inspecting the waveform; vowels are associated with long segments of high amplitude and stop consonants are formed by silent segments ending with a small burst [134]. Nevertheless, speech recognition benefits from access to semantic features [67, 252]. Thus, there is not always a clear-cut distinction between what constitutes a semantic task and what does not.

1.4.2 *Semantic representations*

The tasks discussed above are typically solved with supervised learning where the objective provide some *relative* measure of how well the model encodes semantic information. For unsupervised learning, the ability to learn semantic features must be evaluated differently.

Semantic word relations have been used to evaluate text-based word embeddings [196]. Such relations are defined in a simple arithmetic form (e.g., `paris - france + italy = rome`). This evaluation format does not translate well to speech processing where word boundaries are not easily obtained and word segments have differing length. Thus, the amount of work that employ similar evaluation for speech representations is limited [49]. However, the notion that utterances with similar meaning can be encoded to represent this similarity in terms of a distance metric is certainly relevant.

At the time of writing, there is no unified approach to measuring such similarities for speech representations. One option is to use forced alignment to extract word segments, and then average over the temporal dimension of each segment and different instances of the same word to obtain a single word embedding [215]. Earlier work has used dynamic time warping (DTW), which allows for measuring the distance between continues-valued vector sequences of differing length, but this is rarely used in more recent work. Furthermore, it is typically used to target local acoustic similarities rather than semantic information [239, 240, 286].

On top of the challenges associated with simply finding a way to measure the distance (or similarity) between utterances, such measures will typically be obfuscated by non-semantic information such as acoustics and speaker identity. Most recent work on representa-

tion learning for speech rely on downstream tasks, such as those discussed above, for measuring the semantic information in speech representations [52, 175, 176].

1.4.3 *A working definition*

The brief review of tasks and representations above does not offer a strict definition of semantic speech processing. And indeed, a strict definition is hard to come by. In the following, semantic speech processing should be taken to include models that learn features related to the meaning of language and its constituent symbols from speech. Often, it is not possible to *directly* identify, extract, or evaluate the existence of such features. Instead, it is necessary to analyze the behavior of the models or probe for semantic content in the learned representations. In any case, semantic features learned by a computational model are only an approximation to the human notion of meaning. Thus, we should not get carried away and stay clear of anthropomorphisms.

2 TECHNICAL BACKGROUND

The papers that form the chapters of this thesis are intended to be self-contained. However, the page limits at the largest conferences on speech processing are quite restrictive, so it is valuable to review important technical concepts in greater depth. This chapter presents connectionist temporal classification (CTC) for speech recognition and the self-supervised wav2vec 2.0 framework which are used and discussed extensively throughout the following chapters. Connectionist temporal classification is used in chapter 4, 5, 7 and 8, and the wav2vec 2.0 framework is used and discussed in chapter 6, 7 and 9. Both are briefly described in some of these chapters. However, the description below will provide a greater level of detail, ideally alleviating any need for consulting the original work and allowing the reader to brush over later descriptions. If the reader is already familiar with the frameworks, this chapter can be skipped.

2.1 CONNECTIONIST TEMPORAL CLASSIFICATION

2.1.1 *Background*

Connectionist temporal classification is a general approach for labeling unsegmented sequences [93]. It was introduced in the context of speech recognition, but has also been applied to optical character recognition [272] and machine translation [233]. Together with attention-based encoder-decoder (AED) models [37] and recurrent neural network transducers [92], it is among the most common approaches to end-to-end speech recognition. The *end-to-end* label should be seen in a historical context. Prior to the adoption of these frameworks, the most common approach was based on hidden Markov models (HMMs). This approach relies on separate pronunciation, acoustic, and language models which are trained individually. In contrast, CTC and other end-to-end models can be trained by minimizing a single loss function with an optimization algorithm in the gradient descent family. However, despite the end-to-end label, these models can often be drastically improved by using a separately trained language model for decoding the output [99, 191].

2.1.2 *Forward-pass and decoding*

The studies presented in this thesis will use a simple greedy decoding mechanism, which is presented here, so language model decod-

ing will not be covered. Consider an acoustic input sequence $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_U)$ and its corresponding label sequence $\mathbf{y} = (y_1, \ldots, y_L)$. In practice, $\mathbf{x}$ is typically a spectrogram, but the raw waveform can also be used. If $\mathbf{x}$ is a spectrogram, each $\mathbf{x}_u$ is spectral feature vector computed with a short-time Fourier transformation. If $\mathbf{x}$ is the waveform, each entry is a scalar value. $\mathbf{y}$ is a sequence of symbols (e.g., character or words) from a finite alphabet, such that $y_u \in \{1, \ldots, K\}$. This set includes a *blank* symbol which is key to learning an implicit alignment between $\mathbf{x}$ and $\mathbf{y}$.

Given a set of paired training examples, we want to learn a function $f : \mathbf{x} \mapsto \mathbf{p}$ where $\mathbf{p} = (\mathbf{p}_1, \ldots, \mathbf{p}_T)$ is a sequence of probability vectors, such that $\mathbf{p}_t \in \mathbb{R}^K$ parameterizes a categorical distribution over the alphabet. The function $f(\cdot)$ is a neural network that can process sequential data. Convolutional, recurrent or transformer-based networks have all proven viable solutions [37, 73, 224]. Regardless of whether the input is a spectrogram or the raw waveform, it is common to configure $f(\cdot)$ such that it produces an output $\mathbf{p}_t$ every 0.02 seconds. Thus, T is proportional to the input length U. As we will see next, it is important that the output resolution is not too low.

In order to obtain an estimate of $\mathbf{y}$ from $\mathbf{p}$, a simple greedy decoding mechanism is used. First, obtain a sequence $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_T)$ by letting $a_t = \arg\max_a p_{t,a}$. The sequence $\mathbf{a}$, referred to as a path, can be seen as a hard alignment between $\mathbf{x}$ and $\mathbf{y}$. Next, apply two simple rules defined by a function $\beta(\cdot)$:

(1) Collapse consecutive integers of the same value to a single entry.

(2) Remove all values corresponding to the blank symbol (-).

That is, $\beta(\mathbf{a}) = \hat{\mathbf{y}}$. For example, if $\mathbf{a}$ corresponds to the character sequence `-h-eell-loo-`, the output sequence will be `hello`. The model is commonly evaluated by computing the normalized edit distance between the target label sequence $\mathbf{y}$ and the estimated label sequence $\hat{\mathbf{y}}$. That is, the word, character, or phoneme error rate depending on the choice of output unit and tokenization.

This decoding mechanism has two implications. First, since consecutive values are collapsed if they take the same value, repeated values in the target (i.e., $y_l = y_{l+1}$) must be modeled by inserting an additional blank symbol. Second, the model output length T must be greater than or equal to the label sequence length L. The exact requirement is that $T \geqslant L + R$ where R is the number of times a character is repeated in the target sequence as this warrants an additional blank token as we just saw. Clearly, the greedy decoding mechanism amounts to a simple fixed algorithm, so we only need to learn $f(\cdot)$.

2.1.3 *Loss function and training*

Consider the target sequence $\mathbf{y}$ and a corresponding valid alignment $\mathbf{a}$. As described above, their relation is defined by $\beta(\mathbf{a}) = \mathbf{y}$. If we could define a one-to-one inverse function (i.e., $\beta^{-1}(\mathbf{y}) = \mathbf{a}$) the network $f(\cdot)$ could be trained by minimizing the cross-entropy loss using $\mathbf{a}$ as the target. However, $\beta^{-1}(\cdot)$ is one-to-many for a given length T, so we need to sum over all possible alignments to obtain a probability for $\mathbf{y}$ under the model

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{a}\in\beta^{-1}(\mathbf{y})} P(\mathbf{a}|\mathbf{x}) \tag{1}$$

$$P(\mathbf{a}|\mathbf{x}) = \prod_{a_t} p_{t,a_t} \ . \tag{2}$$

Here, $\mathbf{a}$ can refer to any path that translates to $\mathbf{y}$, not just the best path as used for decoding above. If the number of possible paths is large, which is often the case, this expression can not be evaluated naively. Thus, the CTC loss is computed with a dynamic programming algorithm inspired by the forward-backward algorithm.

First, consider an extended label sequence $\bar{\mathbf{y}}$, where the targets are interleaved and padded with the aforementioned blank token which have index b. Thus, we has $\bar{\mathbf{y}} = (b, y_1, b, y_2, \ldots, b, y_L, b)$ with length $S = 2L + 1$. This makes it explicit that we allow blank tokens between each symbol. Now, let $\alpha_{s,t}$ denote the forward variable, which is defined as the probability of the first s symbols in the extended label sequence (i.e., $\bar{\mathbf{y}}_{1:s}$) given the first t steps of the output (i.e., $\mathbf{p}_{1:t}$).

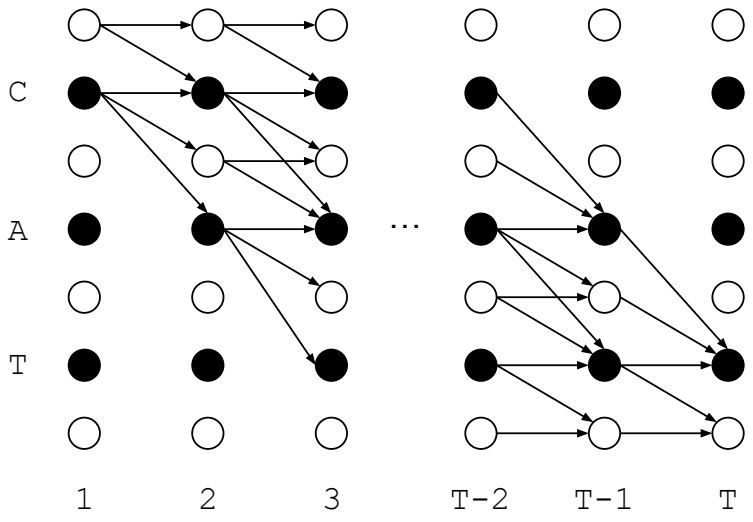


Figure 2: Illustration of how to recursively compute the forward variable $\alpha_{s,t}$ for the CTC loss. Each node corresponds to an $\alpha_{s,t}$. White nodes are blank and black nodes are non-blank. This figure was borrowed from the original CTC paper [93].

This probability can be computed recursively by merging alignments that corresponds to the same label sequence at a given time step t. To see this, consider figure 2, which was borrowed from the original CTC paper [93]. Again, keep in mind that the blanks are optional, so at $t = 1$ there are two possible start states as evident from the first column:

$$\alpha_{1,1} = p_{1,b} \qquad \text{(starting with a blank)} \tag{3}$$

$$\alpha_{2,1} = p_{1,\bar{y}_2} \qquad \text{(starting with the first symbol } \bar{y}_2 = y_1\text{)} \tag{4}$$

Given these initial states, the computation can now proceed recursively. As can be seen from the right side of figure 2, the number of incoming edges depend on whether we are looking at a blank state or a non-blank state. Thus, there are two options as we proceed

$$\alpha_{s,t} = \begin{cases} (\alpha_{s,t-1} + \alpha_{s-1,t-1})\, p_{t,\bar{y}_s} & \text{if } \bar{y}_s = b \text{ or } \bar{y}_s = \bar{y}_{s-2} \\ (\alpha_{s,t-1} + \alpha_{s-1,t-1} + \alpha_{s-2,t-1})\, p_{t,\bar{y}_s} & \text{otherwise} \end{cases} . \tag{5}$$

The first case corresponds to computing $\alpha_{s,l}$ at a blank node, or a node where a letter has been repeated. In case of the latter, which is not illustrated in figure 2, it is not allowed to transition directly from non-blank state to the next non-blank state, and thus, these nodes will also have only two incoming edges.

At the end of recursion, the probability $P(\mathbf{y}|\mathbf{x})$ can be computed by summing the forward variable for the two possible end states $\alpha_{S,L} + \alpha_{S-1,L}$ [98]. However, in order to avoid numerical underflow, $\alpha_{s,l}$ can be rescaled, such that

$$C_l = \sum_s \alpha_{s,l} \tag{6}$$

$$\bar{\alpha}_{s,l} = \alpha_{s,l} / C_l \ , \tag{7}$$

which should be used to replace $\alpha_{s,l}$ on the right-hand side of equation 5. With this rescaling, the log-probability is simply given by

$$\ln P(\mathbf{y}|\mathbf{x}) = \sum_l \ln C_l \ . \tag{8}$$

Since all the operations applied to compute the loss are simple additions and multiplications, it can be differentiated and used to train the function $f(\cdot)$ [98].

How to compute gradients for network parameters will not be covered in detail here. Note, however, that in order to do so, a backward variable $\beta_{s,l}$ is computed in a similar fashion to $\alpha_{s,l}$. In essence, this is obtained by reverting the arrows in figure 2. Indeed, this is why the CTC algorithm cites the forward-backward algorithm as an inspiration. We end up with

$$\frac{\partial P(\mathbf{y}|\mathbf{x})}{\partial p_{l,k}} = -\frac{1}{p_{l,k}^2} \sum_{s \in \{s:\bar{y}_s = k\}} \alpha_{s,l} \beta_{s,l} \ . \tag{9}$$

2.2 WAV2VEC 2.0

2.2.1 *Background*

Wav2vec 2.0 is a framework for self-supervised speech representation learning [13]. Upon pre-training, the model can be fine-tuned for speech recognition with only 10 minutes of labeled data, and achieve a WER of 4.6%/7.9% on the Librispeech clean/other test sets [213]. The frozen model can also be used for feature extraction, as will be the case in chapters 7 and 9 of this thesis. As the name indicates, the model builds on its predecessor wav2vec which is inspired by contrastive predictive coding [206]. Furthermore, the model adapts input masking and a transformer-based architecture with reference to masked language models such as BERT [68].

2.2.2 *Model components*

Wav2vec 2.0 consist of three neural network components: A convolutional feature encoder $f_z(\cdot)$, a transformer-based context network $f_c(\cdot)$ and an step-wise quantization module $g_q(\cdot)$. The architecture and functionality of each of the three modules are described in greater detail below. The original work presents the model in two configurations: BASE and LARGE. The details for these configurations are presented in table 1. To see how the three components relate to each other, we will consider the forward-pass during pre-training. In contrast to the previous section, an explicit sequence subscript will be used here. Thus, the input waveform is denoted by $\mathbf{x}_{1:U} = (x_1, \ldots, x_U)$. We have

$$\mathbf{z}_t = f_z(\mathbf{x}_{u-r:u+r}) \tag{10}$$

$$\mathbf{c}_{1:T} = f_c(m(\mathbf{z}_{1:T})) \tag{11}$$

$$\mathbf{q}_t = g_q(\mathbf{z}_t) \ , \tag{12}$$

where $\mathbf{z}_{1:T}$, $\mathbf{c}_{1:T}$ and $\mathbf{q}_{1:T}$ are vector sequences where $t \propto u$. The receptive field of the feature encoder is $2r + 1$ and the downsampling factor is d. The function $m(\cdot)$ defines a stochastic masking policy which selects some fraction of the vectors $\mathbf{z}_t \in \mathbf{z}_{1:T}$ as starting points from where N consecutive vectors $(\mathbf{z}_t, \ldots, \mathbf{z}_{t+N-1})$ are replaced by a learned vector shared for all masked time-steps. In practice, 6.5% of $\mathbf{z}_{1:T}$ is chosen as starting points and $N = 10$.

When fine-tuned for speech recognition, the original paper adds an output layer on top of $f_c(\cdot)$ which is then fine-tuned, whereas the feature encoder $f_z(\cdot)$ is not updated. When used for feature extraction, it is common to use hidden contextualized representations from the pre-trained frozen $f_c(\cdot)$ [280]. Note that $m(\cdot)$ is removed for downstream feature extraction or replaced by another masking policy for

Feature encoder		Context network	
$L \times$ { 1D-CONV($n = 512, k, s$); LAYER NORMALIZATION; GELU ACTIVATION }		1D-CONV(n=512, k=128, s=1, g=16); GELU ACTIVATION + RESIDUAL; $L \times$ TRANSFORMER ENCODER(d, h)	
BASE/ LARGE	L=7, k=10,3,3,3,3,2,2 s=5,2,2,2,2,2,2	BASE	L=12, d=768, h=8
		LARGE	L=24, d=1024, h=16

Table 1: Specification of the two wav2vec 2.0 configurations presented in the original work [13] and used in later chapters of this thesis. We have n: output channels, k: kernel width, s: stride, g: groups, d: inner dimensionality, and h: number of attention heads.

fine-tuning. Similarly, $g_q(\cdot)$ is only used during training in order to compute a quantized target representation, as we will see below.

FEATURE ENCODER The model takes as input the raw waveform audio with a sample rate of 16kHz. Given that inputs may be several seconds long, it is computationally beneficial to reduce the temporal resolution in the model. This is done by striding with 1D convolutional layers, such that the total downsampling factor is $d = 320$. Thus, the encoder produces an output representation for every 0.02 seconds of input audio. This is similar to the common configuration of CTC-based speech recognition described above, which is indeed used for fine-tuning wav2vec 2.0. Another important reason for downsampling is to reduce the memory footprint of the model. The transformer-based context network would not able to process audio sampled at 16kHz given its $O(T^2)$ memory complexity.

CONTEXT NETWORK Similar to the objective of a masked language model [68], the context network is trained to infer values corresponding to the masked portion of its input $\mathbf{z}_{1:T}$. Thus, this component must use a network architecture that can utilize contextual information. Wav2vec 2.0 rely on a standard transformer encoder [266]. In order to encode relative positional information, a grouped 1D convolutional layer with a residual connection is applied to $\mathbf{z}$ before it is fed the transformer encoder. A grouped convolution is a convolution that has input channels and output channels partitioned into some number of groups. The channels of one input group are only connected to the channels of one output group. As such, it corresponds to having multiple smaller convolutional layers from which the outputs are concatenated.

QUANTIZATION MODULE The quantization module is used to infer discrete targets for the masked time-steps. By learning a target, in-

stead of just reconstructing the input $\mathbf{x}$, the context network does not have to waste capacity on modeling redundant features and noise. For instance, small phase-shifts and amplitude scaling is typically not important for speech recognition. Furthermore, quantization is believed to be helpful when processing noisy speech [11].

In order to quantize $\mathbf{z}_t$, wav2vec 2.0 relies on the Gumbel-Softmax [123, 186] combined with the straight-through estimator [23] which permits differentiable sampling from a categorical distribution. First, $\mathbf{z}_t$ is mapped to $\mathbf{l} \in \mathbb{R}^K$ which is mapped to a probability vector $\mathbf{p} \in \mathbb{R}^K$ via the Gumbel softmax given by

$$p_i = \frac{\exp(l_i + n_i)/\tau}{\sum_k^K \exp(l_k + n_k)/\tau} \tag{13}$$

for $i = 1, \ldots, K$, where τ is a temperature and $\mathbf{n} \in \mathbb{R}^K$ is a random vector with $n_i = -\log(-\log(u_i))$ for $u_i \sim \mathcal{U}(0, 1)$. The vector $\mathbf{n}$ is often called the Gumbel noise. We use $\widetilde{\mathbf{p}}$ to denote the untempered standard softmax (i.e., $\tau = 1$ and $n_i = 0$). Note that taking $\operatorname{argmax}_k(l_k + n_k)$ is equivalent to sample a categorical distribution with parameters $\widetilde{\mathbf{p}}$. However, in practice we cannot obtain a sample with argmax as it is non-differentiable. Given that $\tau \to 0$, $\mathbf{p}$ approaches a one-hot vector, but if τ is too small, training might become unstable. Thus, to realize a one-hot vector, the straight-through estimator is used. Consider the non-differentiable function $\varphi(\mathbf{p})_i = 1$ if $i = \arg\max_k p_k$ and 0 otherwise. The straight-through estimator assumes that the Jacobian $\partial\, \varphi / \partial\, \mathbf{p}$ equals the identity matrix.

Furthermore, wav2vec 2.0 uses product quantization. Thus, $\mathbf{z}_t$ is mapped not to one, but G categorical distributions. The samples from the G distributions are used for a codebook lookup, the resulting vectors are concatenated, and a linear transformation is applied to obtain $\mathbf{q}_t$. Product quantization results in a combinatorial explosion, which allows $\mathbf{q}_t$ to take K^G possible values.

2.2.3 *Pre-training*

The model is trained with a contrastive loss, which is only computed for the masked part of $\mathbf{z}_{1:T}$ in order to avoid a trivial learning problem. For a single time-step t, we have

$$\mathcal{L}_t = -\log\left(\frac{\exp(S_c(\mathbf{c}_t, \mathbf{q}_t))}{\sum_{n \sim \mathcal{D}} \exp(S_c(\mathbf{c}_t, \mathbf{q}_n))}\right) \,, \tag{14}$$

where $S_c(\cdot)$ computes the cosine similarity and $\mathcal{D}$ is a set containing the target index t and distractor indices randomly sampled among other masked time-steps.

Importantly, each $\mathbf{z}_t$, which is used to compute the target $\mathbf{q}_t$, only encodes information from a limited receptive field. If $f_z(\cdot)$ was parameterized with a recurrent neural network or a transformer, it could

(at least in theory) learn to ignore the input and always output a simple repetitive vector sequence. Thus, it would become trivial to infer the correct target for masked time-steps and render the learned representations useless. This is commonly referred to as representation collapse. With a convolutional encoder, this is unlikely to happen, as it can not learn to output a long repetitive pattern given its limited receptive field.

In addition to the contrastive loss, wav2vec 2.0 use a diversity loss to ensure that codebook vectors are evenly used

$$\mathcal{L} = \frac{1}{GK} \sum_{g} -H(\widetilde{\mathbf{p}}_g) \ , \tag{15}$$

where $H(\cdot)$ denotes the entropy. Recall that the tilde accent is used to denote the untempered distribution without Gumbel noise. Thus, the model is encouraged to learn a uniform distribution over the possible target values.

3 MAIN CONTRIBUTIONS

As mentioned above, the chapters of the thesis are self-contained studies. The contributions will therefore be described in the beginning of each. However, one chapter is not written with the next in mind. For this reason, we now go over the organization and the contributions of each chapter with respect to the overall theme of the thesis.

Part ii contains studies on supervised speech processing. It examines some of the problems related to training deep neural networks for tasks that require a semantic understanding of language. Furthermore, these studies motivate the work that is presented in part iii. This part will cover speech recognition, as well as question tracking and symptom detection in medical dialogues.

Chapter 4 examines the ability of the most common end-to-end speech recognition models to utilize context. It is shown that attention-based encoder-decoder (AED) models [37] are more context sensitive than CTC models [93]. Through an ablation, this difference is contributed to the AED model's attention mechanism. An occlusion-based analysis shows that both models rely heavily on contextual information.

Chapter 5 studies spoken question tracking and symptom detection in medical dialogues. The proposed multimodal framework uses the textual representation from a speech recognition model and the corresponding speech segment as input for the two tasks. Using two modalities consistently outperforms either modality on its own, although speech recognition is by far the most important element.

Part iii contains studies on unsupervised representation learning for speech. This area of research has progressed tremendously during the three years of this project. Apart from an overview of this emerging field, the primary contributions in this part lie in analyzing and comparing different approaches to speech representation learning.

Chapter 6 presents a comprehensive overview of unsupervised neural representation learning for speech. Previous work is grouped into self-supervised methods and probabilistic latent variable models and described from a high-level perspective. The description is used to derive a model taxonomy that informs a discussion on the models representational power, learning strategies, and evaluation procedures. This chapter forms the basis for the following three chapters.

Chapter 7 compares contrastive speech representations from the popular wav2vec framework [13, 241]. Specifically, the frozen representations are tested as input for low-resource speech recognition. Representations from wav2vec 2.0 are outperformed by its predecessor on a small 10 minutes subset and generally exhibit unstable training. Principal component analysis is shown to alleviate the training issues and highlights that the representations live in low-dimensional subspace.

Chapter 8 proposes a new hierarchical latent variable model for speech inspired by the Clockwork VAE [238]. The model is benchmarked against other latent variable models and autoregressive models for speech. It is shown that a hierarchy of latent variables improves the likelihood of the model. Finally, the latent space of the models are analyzed in terms of phonetic content.

Chapter 9 tries to answer if self-supervised learning has matured enough to stand in for speech recognition in spoken language understanding. Representations from wav2vec 2.0 are systematically compared to speech recognition transcripts on four spoken language understanding tasks. On three classifications tasks, the speech representations are highly competitive, and often outperform the transcripts. On a speech translation task, the speech recognition transcripts are still a better choice.

Part II

SUPERVISED LEARNING

4

DO END-TO-END SPEECH RECOGNITION MODELS CARE ABOUT CONTEXT?

ABSTRACT

The two most common paradigms for end-to-end speech recognition are connectionist temporal classification (CTC) and attention-based encoder-decoder (AED) models. It has been argued that the latter is better suited for learning an implicit language model. We test this hypothesis by measuring temporal context sensitivity and evaluate how the models perform when we constrain the amount of contextual information in the audio input. We find that the AED model is indeed more context sensitive, but that the gap can be closed by adding self-attention to the CTC model. Furthermore, the two models perform similarly when contextual information is constrained. Finally, in contrast to previous research, our results show that the CTC model is highly competitive on WSJ and LibriSpeech without the help of an external language model.

4.1 INTRODUCTION

Connectionist temporal classification (CTC) [93] and attention-based encoder-decoder (AED) models [15, 37] are arguably the most popular choices for end-to-end automatic speech recognition (E2E ASR). However, it has been unclear if CTC and AED models process speech in qualitatively different ways. The use of sentence-level context is important for human speech perception [117], but has not been studied for ASR. Previous research has claimed that AED models learn a better implicit language model given enough training data [20]. Furthermore, comparisons of the two models have suggested that CTC models are inferior without the help of an external language model [20, 223], which leads to the hypothesis that CTC models are incapable of exploiting long temporal dependencies.

We study how the two E2E ASR models utilize temporal context. For this purpose, we consider first-order derivatives [70, 86] and the occlusion of input features [284]. While these methods have been frequently used to analyze natural language processing models [6, 7, 169, 170] their application to speech recognition has been limited [24, 158].

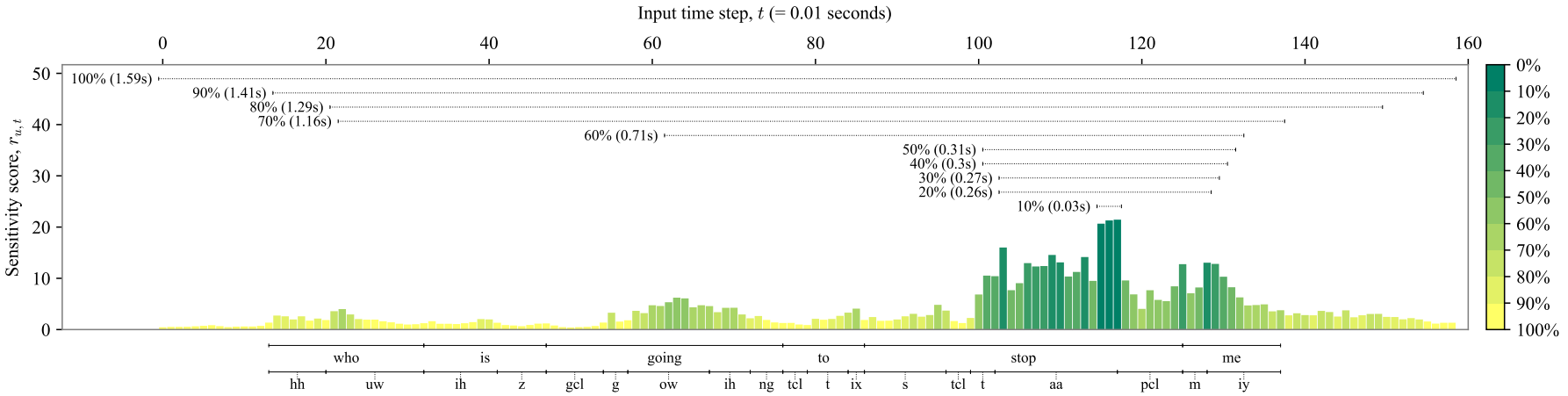


Figure 3: Sensitivity scores for the character "p" in the correctly predicted sentence "who is going to stop me" by the CTC model trained on LibriSpeech. Hand-annotated word and phone alignments from the TIMIT dataset are shown in the bottom. The temporal spans corresponding to different levels of accumulated sensitivity are shown in the top. By averaging these across all non-blank character predictions in a test set, we obtain a measurement of the model's context sensitivity.

1. Through a derivative-based sensitivity analysis we show that the AED model is more context sensitive than the CTC model. Our ablation study attributes this difference to the attention-mechanism which closes the gap when applied to the CTC model.

2. Although the AED model is more context sensitive than the CTC model without an attention-mechanism, we find that the two models perform similarly when contextual information is constrained by occluding surrounding words in the input audio.

3. In contrast to previous comparisons, we show that the CTC model is highly competitive with the AED model without the help of an external language model. Using a deep and densely connected architecture, both models reach a new E2E state-of-the-art on the WSJ task.

4.2 END-TO-END SPEECH RECOGNITION

4.2.1 *Connectionist Temporal Classification Models*

Given a sequence of real valued input vectors $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_T)$, CTC models compute an output sequence $\hat{\mathbf{y}} = (\hat{\mathbf{y}}_1, ..., \hat{\mathbf{y}}_U)$, where each $\hat{\mathbf{y}}_u$ is a categorical probability distribution over the target character set. Apart from the letters a-z, white-space and apostrophe ('), the character set also includes the special blank token (-). The input and output lengths, T and U, are related by $U = \lceil \frac{T}{R} \rceil$ where R is a constant reduction factor achieved by striding or stacking adjacent temporal representations. In this study, we never use an external language model. Instead, we rely on a simple greedy decoder $\beta(\cdot)$ that collapses repeated characters and removes blank tokens (e.g., *-c-aatt-* $\mapsto$ *cat*). The $\beta(\cdot)$ function operates on the predicted alignment path $\hat{\mathbf{q}} = (\hat{q}_1, ..., \hat{q}_U)$ obtained by letting $\hat{q}_u = \arg\max_q \hat{y}_{u,q}$.

This decoding mechanism results from the CTC loss function. The loss is computed by summing the probability of all alignment paths $\mathbf{q} = (q_1, ..., q_U)$ that translate to the target sequence $\mathbf{y}$. The probability of a single path is given by:

$$P(\mathbf{q}|\mathbf{x}) = \prod_{u=1}^{U} \hat{y}_{u,q_u} \tag{16}$$

Given the set of paths $\{\mathbf{q} \mid \beta(\mathbf{q}) = \mathbf{y}\} = \beta^{-1}(\mathbf{y})$ that translate to a given target transcript, the total probability is:

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{q} \in \beta^{-1}(\mathbf{y})} P(\mathbf{q}|\mathbf{x}) \tag{17}$$

The loss is simply $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \ln(P(\mathbf{y}|\mathbf{x}))$ which can be computed efficiently with dynamic programming [93].

4.2.2 *Attention-based Encoder-Decoder Models*

AED models first encode the input $\mathbf{x}$ to a sequence of vectors $\mathbf{h} = (\mathbf{h}_1, ..., \mathbf{h}_U) = \textsc{encode}(\mathbf{x})$ which is passed to an autoregressive decoder function $\textsc{decode}(\cdot)$. We reuse U to denote the length of $\mathbf{h}$ to emphasize that, as with CTC models, it is defined by a constant reduction factor R. However, AED models are typically robust to a high reduction factor ($R \leqslant 2^4$) compared to CTC models ($R \leqslant 2^1$) [20]. Operating at a lower temporal resolution should make it easier for recurrent encoder layers (section 4.2.3) to pass information across longer time spans.

Roughly speaking, we could write the decoder as $\hat{\mathbf{y}}_k = \textsc{decode}(\mathbf{h}, \hat{\mathbf{y}}_{k-1}, \mathbf{s}_{k-1}, \mathbf{a}_{k-1})$ where $\hat{\mathbf{y}}_k$ is a probability distribution over characters, $\mathbf{s}_k$ is the decoder state and $\mathbf{a}_k$ is the attention vector. Unlike CTC models, there are no repeated characters or blank tokens to interleave the final predictions. Thus, given the same output sequence, we have $K \leqslant U$. As with the encoder, lower temporal resolution between decoder steps could make it easier to pass information between predictions.

Emphasizing more detail, we split the $\textsc{decode}(\cdot)$ function into the following sequence of computations:

$$\mathbf{s}_k = \textsc{recurrent}(\mathbf{s}_{k-1}, [\Phi(\hat{\mathbf{y}}_{k-1}); \mathbf{a}_{k-1}]) \tag{18}$$

$$\mathbf{a}_k = \textsc{attend}(\mathbf{s}_k, \mathbf{h}) \tag{19}$$

$$\hat{\mathbf{y}}_k = \textsc{predict}(\mathbf{a}_k) \tag{20}$$

Here $[\cdot\,;\cdot]$ denotes the concatenation of two vectors and $\Phi(\cdot)$ is a non-differentiable embedding lookup.[1] The $\textsc{recurrent}(\cdot)$ function can take the form of any recurrent neural network architecture. We use a single LSTM [109] cell for all our experiments. The $\textsc{predict}(\cdot)$ function is a single fully-connected layer followed by the softmax function. The following steps define the $\textsc{attend}(\cdot)$ function:

$$e_{k,u} = \mathbf{v}^\top \tanh(\mathbf{W}_s \mathbf{s}_k + \mathbf{W}_h \mathbf{h}_u) \tag{21}$$

$$\alpha_{k,u} = \frac{\exp(e_{k,u})}{\sum_{u'=1}^{U} \exp(e_{k,u'})} \tag{22}$$

$$\mathbf{c}_k = \sum_{u=1}^{U} \alpha_{k,u} \mathbf{h}_u \tag{23}$$

$$\mathbf{a}_k = \tanh(\mathbf{W}_a [\mathbf{c}_k; \mathbf{s}_k]) \tag{24}$$

Where $\mathbf{v}$, $\mathbf{W}_s$, $\mathbf{W}_h$ and $\mathbf{W}_a$ are trainable parameters. The computation of the energy coefficient $e_{k,u}$ is taken from [14]. Note that each

1 The lookup is not captured by the gradient-based sensitivity analysis presented in 4.3.1.

energy coefficient, and thus each attention weight $\alpha_{k,u}$, is computed identically for all encoder representations $\mathbf{h}_u$. Unlike recurrent network connections, combining information from time steps far apart does not require propagating the information through a number of computations proportional to the distance between the time steps.

Thus, we have highlighted three components that could make AED models more context sensitive: (I) Encoder resolution, (II) decoder resolution and (III) the attention-mechanism.

4.2.3 *Encoder Architecture*

Whereas the main contribution of the CTC framework is the loss function, the AED model relies on a more complex architecture that allows it to be trained with a simple cross-entropy loss. To see this, note that the CTC forward pass can be stated as a subset of the functions introduced in section 4.2.2:

$$\mathbf{h} = \textsc{encode}(\mathbf{x}) \tag{25}$$

$$\hat{\mathbf{y}}_u = \textsc{predict}(\mathbf{h}_u) \tag{26}$$

As in previous work, we use convolutions followed by a sequence of bidirectional recurrent neural networks [2, 214, 287]. Our final encoder has 10 bidirectional LSTM layers with skip-connections inspired by [115]. The outputs of the forward and backward cells are summed after each LSTM layer. Default is $R = 2$ for CTC and $R = 4$ for AED. See figure 4.

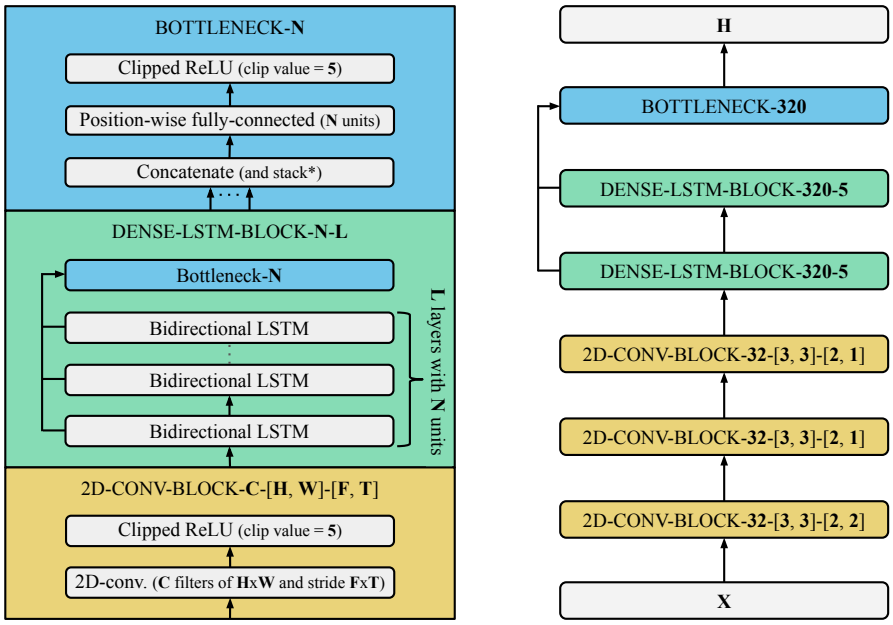


Figure 4: Default encoder architecture used for both CTC and AED models. * Only applied in the bottleneck layer of the first dense LSTM block for the AED model to achieve $R = 4$.

4.3 METHOD

We used two different approaches for analyzing temporal context utilization of the two E2E ASR models. The derivative-based sensitivity analysis (4.3.1) can be used to compare a set of models on any dataset. However, as we will see, there is no guarantee that the differences found with this approach translate to better performance. The occlusion-based analysis (4.3.2) allows us to evaluate how the models respond when we remove temporal context. This measure is easy to interpret and can be used to directly asses the importance of temporal context, but requires hand-annotated word-alignments which are rarely available in publicly available datasets.

4.3.1 *Derivative-based Sensitivity Analysis*

We define a sequence of sensitivity scores $\mathbf{r}_k$ ($\mathbf{r}_u$ for CTC models) across the temporal dimension of the input space for each predicted character. Let F be number of spectral input features and Q the size of the output character set:

$$r_{k,t} = \sum_{q=1}^{Q} \sum_{f=1}^{F} \left| \frac{\partial \hat{y}_{k,q}}{\partial x_{t,f}} \right| \tag{27}$$

An example is shown in figure 3. Our goal is to measure the dispersion of these scores across the input time steps. We do so by summing the scores from largest to smallest and measure the temporal span of the scores accumulated for a certain percentage of the total sensitivity. For example, the set of scores needed to account for 10% of the total sensitivity may be $\{r_{k,3}, r_{k,7}, r_{k,8}, r_{k,10}\}$. The temporal span would then be $10 - 3 = 7$ time steps corresponding to 0.07 seconds. We take the mean of this span for a fixed percentage across all character predictions in a given data set to summarize the temporal context sensitivity of a model. This allows us to evaluate how the sensitivity disperses as we increase the accumulated percentage. A higher dispersion of sensitivity scores equals a higher context sensitivity. The derivative-based measure considers a linearization of the models and, thus, does not capture non-linear effects.

4.3.2 *Occlusion-based Analysis*

To directly test the dependence on contextual information, we use hand-annotated word-alignments to systematically occlude context. Given a word w_t, we test how well a model recognizes the word given different levels of context. That is, we crop out the audio segment corresponding to $w_{t-C}, ..., w_{t+C}$ where C is the maximum number

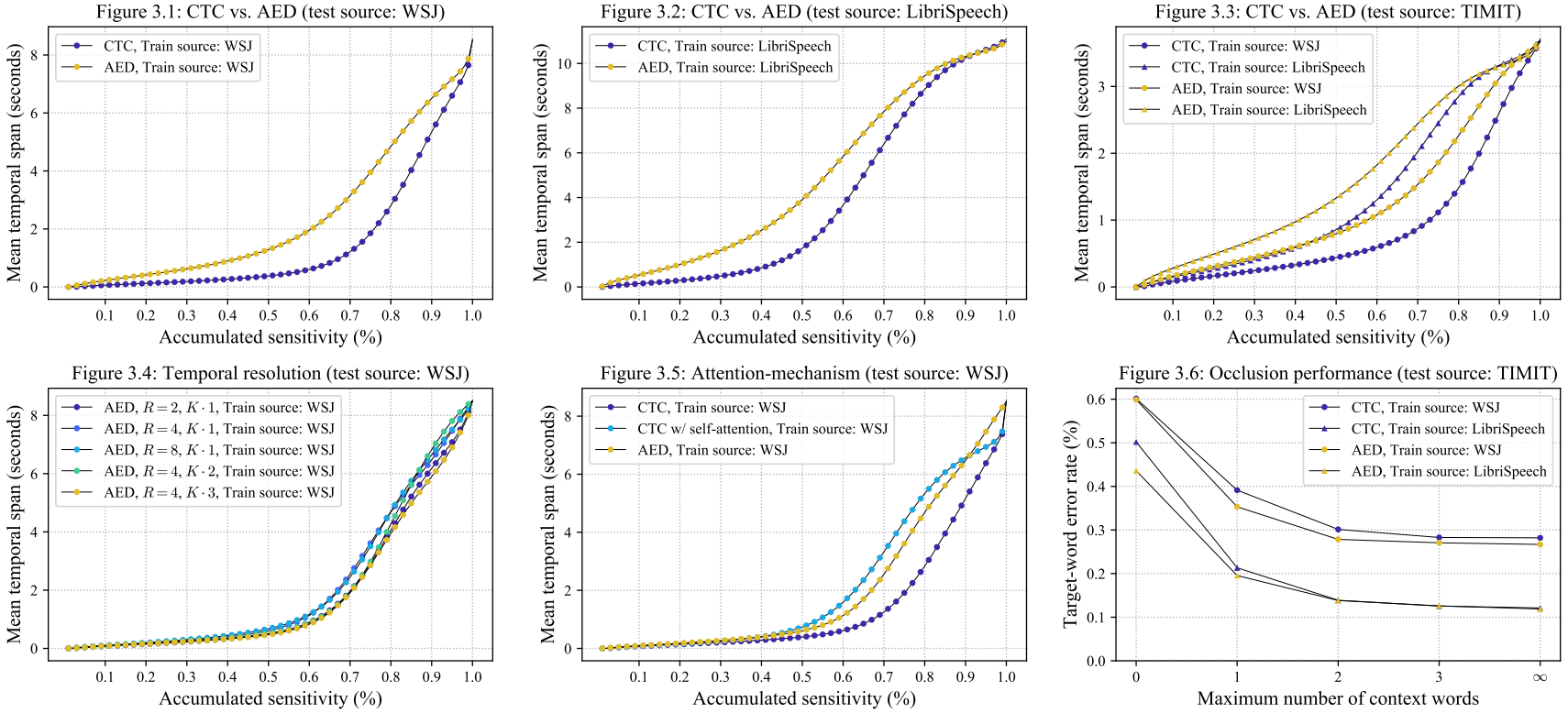


Figure 5: Sensitivity analysis (figures 3.1-3.5) and occlusion-based analysis (figure 3.6). See corresponding subsections.

of context words visible on each side.[2] If the target word w_t is in the predicted sequence, we accept the hypothesis. To avoid ambiguous situations where the target word is identical to one of the 2C context words, we only make use of sentences that consist of a sequence of unique words.

4.4 EXPERIMENTS

4.4.1 *Data and Training*

We trained the models on the Wall Street Journal CSR corpus (WSJ) [220] and the LibriSpeech ASR corpus [213]. WSJ contains approximately 81 hours of read newspaper articles and LibriSpeech contains 960 hours of audio book samples. We used 80-dimensional log-mel spectrograms as input. The models were trained for 600 epochs on WSJ and 120 epochs on LibriSpeech. We used Adam [146] with a fixed learning rate of $3 \cdot 10^{-4}$ for the first 100 epochs on WSJ and 20 epochs on LibriSpeech, before annealing it to 1/6 of its original size. We used dropout after each convolutional block [260] and each bidirectional LSTM layer [151]. The dropout rate was set to 0.10 for models trained on LibriSpeech and 0.40 for WSJ. Similar to [241], we constructed batches of similar length samples, such that one batch consisted of up to 320 seconds of audio and contained a variable number of samples. For the AED model, we used teacher-forcing with a 10% sampling rate.

For the occlusion-based analysis, we considered the hand-annotated word-alignments from the TIMIT dataset [87]. We excluded all sentences repeated by multiple speakers in order to avoid biasing the results towards certain sentence constructions (i.e., we only use the SI-files of the TIMIT dataset).

4.4.2 *ASR results*

We compare the default configuration of our CTC and AED models trained on WSJ and LibriSpeech to other notable E2E ASR models in table 3 and 2. Both the CTC and AED model compare favorably to more sophisticated approaches on WSJ. On LibriSpeech, our models do not perform as well as larger models, but are still on par with models of comparable size from [120] which is the same model as in [214] at smaller scale. The slightly worse performance of the AED model on LibriSpeech can be attributed to longer sentences which have a tendency to destabilize training. Similar issues have been reported in prior work [20, 37].

2 We also add the silence from the start and end of the original sentence to the audio segment as it improves model performance.

Model	eval92	type
Chorowski & Jaitly, 2016 [46]	10.60	AED
Zhang et al., 2017 [287]	10.53	AED
Chan et al., 2016 [38]	9.6	AED
Sabour et al., 2018 [232]	9.3	AED
Our work:		
Deep LSTM	9.25	CTC
Deep LSTM	9.25	AED

Table 2: Word error rates on the eval92 test set of WSJ. None of the above use an external language model.

Model	clean	other	type	params
Li et al., 2019 [168]	3.86	11.95	CTC	333 M
Kim et al., 2019 [145]	3.66	12.39	AED	~320 M
Park et al., 2019 [214]	2.80	6.80	AED	~280 M
Irie et al., 2019 [120]				
Small - Grapheme	7.9	21.3	AED	7 M
Small - Word-piece	6.1	16.4	AED	20 M
Medium - Grapheme	5.6	15.8	AED	35 M
Medium - Word-piece	5.0	14.1	AED	60 M
Our work:				
Deep LSTM	5.13	16.03	CTC	17.7 M
Deep LSTM	5.45	17.05	AED	19.8 M

Table 3: Word error rates on the clean and other test sets of LibriSpeech. None of the above use an external language model.

4.4.3 *CTC vs. AED*

As hypothesized, figure 3.1 and 3.2 reveal that our AED models utilized a larger temporal context than the CTC models based on the sensitivity scores. The trend was consistent across all levels of accumulated sensitivity scores. In figure 3.3, we see the same pattern when evaluated on the TIMIT dataset which will be used for the occlusion-based analysis.

4.4.4 *Temporal resolution*

We trained the AED model with three different temporal encoder resolutions $R = 2, 4, 8$ on the WSJ dataset. R was configured by increasing stride in each of the three convolutional layers. As seen in figure 3.4, encoder resolution had no impact on context sensitivity.

To test decoder resolution, we interleaved the target transcript with one or two redundant blank tokens to effectively increase the target length to $K \cdot 2$ or $K \cdot 3$. Figure 3.4 shows that decoder resolution had no impact on context sensitivity.

4.4.5 *Attention-mechanism*

To test how the attention-mechanism affects context sensitivity, we incorporated the $\textsc{attend}(\cdot)$ function in the CTC architecture. Instead of passing $\mathbf{h}_u$ directly to $\textsc{predict}(\cdot)$, we first applied self-attention:

$$\hat{\mathbf{y}}_u = \textsc{predict}(\textsc{attend}(\mathbf{h}_u, \mathbf{h})) \tag{28}$$

We trained this model on the WSJ dataset and compared it to the AED model and the CTC model without attention in figure 3.5. The attention-mechanism closed the gap in context sensitivity between the two models. Thus, the difference found in sections 4.4.3 is likely a result of this architectural component that can be easily incorporated in a CTC model. However, a large U results in high memory consumption. Therefore, we used a smaller model where the two dense LSTM blocks are replaced by three LSTM layers with 128 units for the experiments shown in figures 3.4 and 3.5.

4.4.6 *Occlusion performance*

Figure 3.6 shows how model performance is affected under different context constraints. We see that both the CTC and AED model suffered severely when contextual information was completely removed. The models came close to optimal performance when approximately three words were allowed on each side of the target word. Thus, temporal context is an important factor for both models. This result aligns

well with the common n-gram size (3-4) when decoding with the help of a statistical language model [46, 183, 283].

Based on the results in section 4.4.3, we would expect that the AED models rely more on the temporal context than the CTC model. However, we do not see such a trend in figure 3.6. Indeed, there was no pronounced or consistent difference between the two models regardless of training source. This result implies that the architectural differences between the AED and CTC models do not necessarily translate to a performance difference. It may be that the AED model included more evidence from context than the CTC model, but the results in figure 3.6 indicate that this did not add any additional value in terms of lowering word error rate.

4.5 CONCLUSIONS

We show that AED models are generally more context sensitive than CTC models and that this difference is largely explained by the attention-mechanism of AED models. Adding a self-attention layer to the CTC model bridges the gap between the models. Analyzing performance by constraining temporal context, we also find that the initial difference between the two models is not crucial in terms of word error rate performance, although both models rely heavily on context for optimal performance. Our experiments on WSJ and LibriSpeech show that CTC models are capable of delivering state-of-the-art results on par with AED models without an external language model. Because of its simplicity and more stable training, CTC is our preferred E2E ASR framework.

5

MULTIQT: MULTIMODAL LEARNING FOR REAL-TIME QUESTION TRACKING IN SPEECH

ABSTRACT

We address a challenging and practical task of labeling questions in speech in real time during telephone calls to emergency medical services in English, which embeds within a broader decision support system for emergency call-takers. We propose a novel multimodal approach to real-time sequence labeling in speech. Our model treats speech and its own textual representation as two separate modalities or views, as it jointly learns from streamed audio and its noisy transcription into text via automatic speech recognition. Our results show significant gains of jointly learning from the two modalities when compared to text or audio only, under adverse noise and limited volume of training data. The results generalize to medical symptoms detection where we observe a similar pattern of improvements with multimodal learning.

5.1 INTRODUCTION

Our paper addresses the challenge of learning to discover and label questions in telephone calls to emergency medical services in English. The task is demanding in two key aspects:

1. **Noise:** A typical phone call to an emergency medical service differs significantly from data within most standard speech datasets. Most importantly, emergency calls are noisy by nature due to very stressful conversations conveyed over poor telephone lines. Automatic speech recognition (ASR) and subsequent text processing quickly becomes prohibitive in such noisy environments, where word error rates (WER) are significantly higher than for standard benchmark data [97]. For this reason, we propose a sequence labeler that makes use of two modalities of a phone call: audio and its transcription into text by utilizing an ASR model. Hereby we create a multimodal architecture that is more robust to the adverse conditions of an emergency call.

2. **Real-time processing:** Our model is required to work incrementally to discover questions in real time within incoming streams of audio in order to work as a live decision support system. At runtime, no segmentation into sub-call utterances such as phrases or sentences is easily available. The lack of

segmentation coupled with the real-time processing constraint makes it computationally prohibitive to discover alignments between speech and its automatic transcription. For these reasons, we cannot utilize standard approaches to multimodal learning which typically rely on near-perfect cross-modal alignments between short and well-defined segments [16].

CONTEXT AND RELEVANCE. Learning to label sequences of text is one of the more thoroughly explored topics in natural language processing. In recent times, neural networks are applied not only to sequential labeling like part-of-speech tagging [222] or named entity recognition [184], but also to cast into a labeling framework otherwise non-sequential tasks such as syntactic parsing [90, 254].

By contrast, assigning labels to audio sequences of human speech is comparatively less charted out. When addressed, speech labeling typically adopts a solution by proxy, which is to automatically transcribe speech into text, and then apply a text-only model [80, 199, 255]. The challenge then becomes not to natively label speech, but to adapt the model to adverse conditions of speech recognition error rates. Such models typically feature in end-to-end applications such as dialogue state tracking [105, 226]. Recent advances in end-to-end neural network learning offer promise to directly label linguistic categories from speech alone [88]. From another viewpoint, multimodal learning is successfully applied to multimedia processing where the modalities such as text, speech, and video are closely aligned. However, contributions there typically feature classification tasks such as sentiment analysis and not finer-grained multimedia sequence labeling [281].

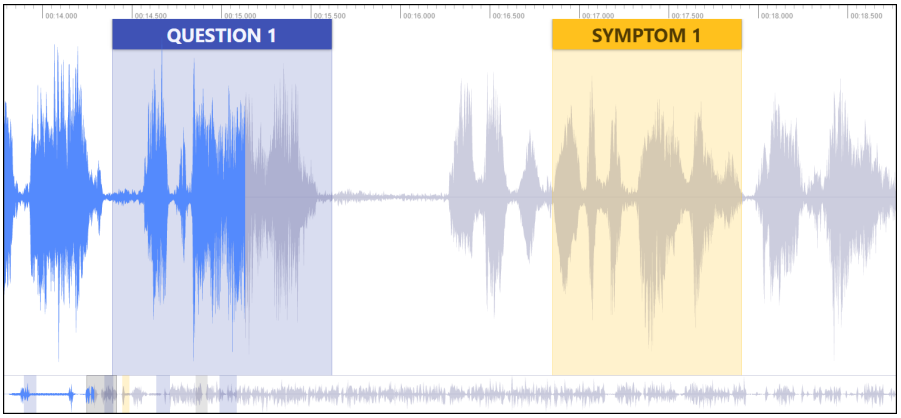


Figure 6: A speech sequence from our phone call dataset. Two audio segments are highlighted: a question (in blue) and a reported symptom (in yellow).

OUR CONTRIBUTIONS. We propose a novel neural architecture to incrementally label questions in speech by learning from its two modalities or views: the native audio signal itself and its transcription into noisy text via ASR.

1. Our model utilizes the online temporal alignment between the input audio signal and its raw ASR transcription. By taking advantage of this fortuitous real-time coupling, we avoid having to learn the multimodal alignment over the entire phone call and its transcript, which would violate the real-time processing constraint that is crucial for decision support.

2. We achieve consistent and significant improvements from learning jointly from the two modalities compared to ASR transcriptions and audio only. The improvements hold across two inherently different audio sequence labeling tasks.

3. Our evaluation framework features a challenging real-world task with noisy inputs and real-time processing requirements. Under this adversity, we find questions and medical symptoms in emergency phone calls with high accuracy. Our task is illustrated in Figure 6.

5.2 MULTIMODAL SPEECH LABELING

We define the multimodal speech labeler MultiQT as a combination of three neural networks that we apply to a number of temporal input modalities. In our case, we consider speech and associated machine transcripts as the separate modalities or views. The model is illustrated in Figure 7.

To obtain temporal alignment between speech and text, we propose a simple approach that uses the output of an ASR system as the textual representation. Here, we take the ASR to be a neural network trained with the connectionist temporal classification (CTC) loss function [93]. Given audio, it produces a temporal softmax of length T_s with a feature dimension defined as a categorical distribution, typically over characters, words or subword units, per timestep.

We refer to a sequence of input representations of the audio modality as $(\mathbf{x}_a^{(t)})_{t\in[1..T_a]}$ and of the textual modality as $(\mathbf{x}_s^{(t)})_{t\in[1..T_s]}$. From the input sequences we compute independent unimodal representations denoted by $\mathbf{z}_a^{(t)}$ and $\mathbf{z}_s^{(t)}$ by applying two unimodal transformations denoted by f_a and f_s, respectively. Each of these transformations is parameterized by a convolutional neural network with overall

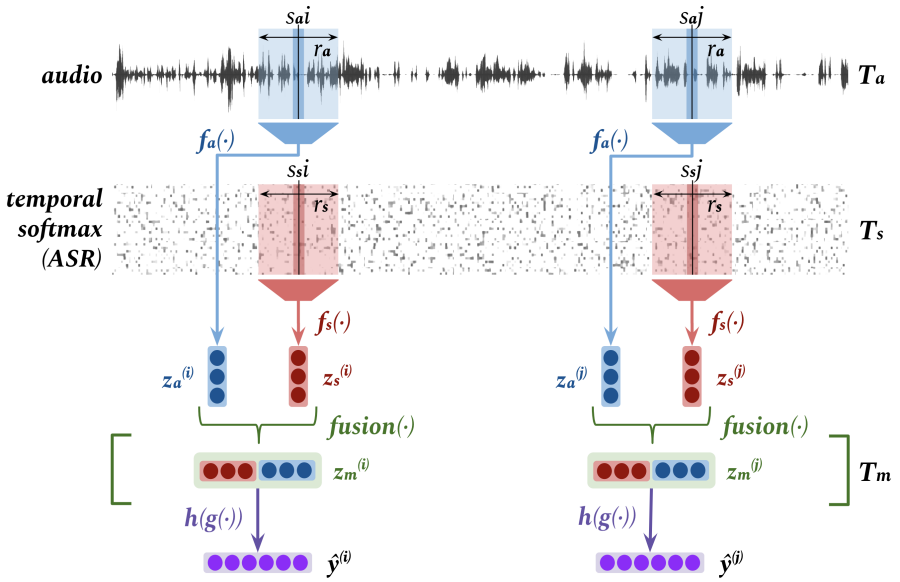


Figure 7: MultiQT model illustration for two timesteps i and j. We depict the convolutional transformations f_a and f_s of the audio and character temporal softmax inputs into the respective modality encodings $\mathbf{z}_a^{(i)}$ and $\mathbf{z}_s^{(i)}$, along with the corresponding receptive fields and strides: r_a, s_a and r_s, s_s. The convolutions are followed by multimodal fusion and finally dense layers g and h to predict the question labels $\hat{\mathbf{y}}^{(i)}$ and $\hat{\mathbf{y}}^{(j)}$.

temporal strides s_a and s_s and receptive fields r_a and r_s. With T_m as length of the resulting unimodal representations:

$$\begin{aligned} \mathbf{z}_a^{(t)} &= f_a\left(\left(\mathbf{x}_a^{(i)}\right)_{i=s_a t - r_{a,l}}^{s_a t + r_{a,r}}\right) \\ \mathbf{z}_s^{(t)} &= f_s\left(\left(\mathbf{x}_s^{(i)}\right)_{i=s_s t - r_{s,l}}^{s_s t + r_{s,r}}\right), \end{aligned} \tag{29}$$

for $t \in [1..T_m]$, where $r_{a,l}$, $r_{a,r}$, $r_{s,l}$ and $r_{s,r}$ are the left and right half receptive fields of f_a and f_s, respectively. For f_a, $r_{a,l} = \lfloor (r_a - 1)/2 \rfloor$ and $r_{a,r} = \lceil (r_a - 1)/2 \rceil$ and similarly for f_s. For $i < 1$ and $i > T_a$ we define $\mathbf{x}_a^{(i)}$ and $\mathbf{x}_s^{(i)}$ by zero padding, effectively padding with half the receptive field on the left and right sides of the input. This then implies that $T_m = \lfloor T_a / s_a \rfloor = \lfloor T_s / s_s \rfloor$ which constrains the strides according to T_a and T_s and functions as "same padding". This lets us do convolutions without padding the internal representations for each layer in the neural networks, which in turn allows for online streaming.

To form a joint multimodal representation from $\mathbf{z}_a^{(t)}$ and $\mathbf{z}_s^{(t)}$ we join the representations along the feature dimension. In the multimodal learning litterature such an operation is sometimes called fusion [281]. We denote the combined multimodal representation by

$\mathbf{z}_m^{(t)}$ and obtain it in a time-binded manner such that for a certain timestep $\mathbf{z}_m^{(t)}$ only depends on $\mathbf{z}_a^{(t)}$ and $\mathbf{z}_s^{(t)}$,

$$\mathbf{z}_m^{(t)} = \text{fusion}\left(\mathbf{z}_a^{(t)}, \mathbf{z}_s^{(t)}\right). \tag{30}$$

In our experiments fusion$(\cdot)$ either denotes a simple concatenation, $[\mathbf{z}_a^{(t)}; \mathbf{z}_s^{(t)}]$, or a flattened outer product, $[1\ \mathbf{z}_a^{(t)}] \otimes [1\ \mathbf{z}_s^{(t)}]$. The latter is similar to the fusion introduced by Zadeh et al. [281], but we do not collapse the time dimension since our model predicts sequential labels.

Finally, $\mathbf{z}_m^{(t)}$ is transformed before projection into the output space:

$$\mathbf{z}_y^{(t)} = g\left(\mathbf{z}_m^{(t)}\right), \tag{31}$$

$$\hat{\mathbf{y}}^{(t)} = h\left(\mathbf{z}_y^{(t)}\right), \tag{32}$$

where g is a fully connected neural network and h is a single dense layer followed by a softmax activation such that $\hat{\mathbf{y}}^{(t)} \in \mathbb{R}^K$ is a vector of probabilities summing to one for each of the K output categories. The predicted class is $\arg\max(\hat{\mathbf{y}}^{(t)})$.

5.2.1 *Objective functions*

In general, the loss is defined as a function of all learnable parameters Θ and is computed as the average loss on M examples in a mini-batch. We denote by $\{\mathcal{X}_a, \mathcal{X}_s\}$ a dataset consisting of N pairs of input sequences of each of the two modalities. As short-hand notation, let $\mathbf{X}_a^{(n)}$ refer to the n'th audio sequence example in $\mathcal{X}_a$ and similarly for $\mathbf{X}_s^{(n)}$. The mini-batch loss is then

$$\mathcal{L}\left(\Theta; \left\{\mathbf{X}_a^{(n)}, \mathbf{X}_s^{(n)}\right\}_{n\in\mathcal{B}_i}\right) = \frac{1}{M}\sum_{n\in\mathcal{B}_i} \mathcal{L}^{(n)}\left(\Theta; \mathbf{X}_a^{(n)}, \mathbf{X}_s^{(n)}\right), \tag{33}$$

where $\mathcal{B}_i$ is an index set uniformly sampled from [1..N] which defines the i'th batch of size $|\mathcal{B}_i| = M$.

The loss for each example, $\mathcal{L}^{(n)}$, is computed as the time-average of the loss per timestep,

$$\mathcal{L}^{(n)}\left(\Theta; \mathbf{X}_a^{(n)}, \mathbf{X}_s^{(n)}\right) = \frac{1}{T}\sum_{t=1}^{T} \mathcal{L}^{(n,t)}\left(\Theta; \mathbf{X}_a^{(n,\mathbf{t}_a)}, \mathbf{X}_s^{(n,\mathbf{t}_s)}\right), \tag{34}$$

where $\mathbf{t}_a = [s_a t - r_{a,l} \,..\, s_a t + r_{a,r}]$ and similarly for $\mathbf{t}_s$ since the dependency of the loss per timestep is only on a limited timespan of the input. The loss per timestep is defined as the categorical cross-entropy loss between the softmax prediction $\hat{\mathbf{y}}^{(t)}$ and the one-hot encoded ground truth target $\mathbf{y}^{(t)}$,

$$\mathcal{L}^{(n,t)}\left(\Theta; \mathbf{X}_a^{(n,\mathbf{t}_a)}, \mathbf{X}_s^{(n,\mathbf{t}_s)}\right) = \sum_{k=1}^{K} y_k^{(t)} \log(\hat{y}_k^{(t)}).$$

Label	Description	Example	Count	Fraction
Q1	Question about the address of the incident.	*What's the address?*	663	26.3%
Q2	Initial question of the call-taker to begin assessing the situation.	*What's the problem?*	546	21.6%
Q3	Question about the age of the patient.	*How old is she?*	537	21.3%
Q4	All questions related to patient's quality of breathing.	*Is she breathing in a normal pattern?*	293	11.6%
Q5	All question about patient's consciousness or responsiveness.	*Is he conscious and awake?*	484	19.2%

Table 4: Explanation and prevalence of the questions used for the experiments.

The full set of learnable parameters Θ is jointly optimized by mini-batch stochastic gradient descent.

5.2.2 *Multitask objective*

In addition to the loss functions defined above, we also consider multitask training. This has been reported to improve performance in many different domains by including a suitably related auxiliary task [27, 187].

For the task of labelling segments in the input sequences as pertaining to annotations from among a set of $K-1$ positive classes and one negative class, we propose the auxiliary task of binary labelling of segments as pertaining to either the negative class or any of the $K-1$ positive classes. For question tracking, this amounts to doing binary labelling of segments that are questions of any kind. The hope is that this will make the training signal stronger since the sparsity of each of the classes, e.g. questions, is reduced by collapsing them into one shared class.

We use the same loss function as above, but with the number of classes reduced to $K = 2$. The total multitask loss is a weighted sum of the K-class loss and the binary loss:

$$\mathcal{L}_{\text{MT}}^{(n,t)} = \beta \mathcal{L}_{\text{binary}}^{(n,t)} + (1-\beta)\mathcal{L}^{(n,t)}. \tag{35}$$

The tunable hyperparameter $\beta \in [0, 1]$ interpolates the task between regular K-class labeling for $\beta = 0$ and binary classification for $\beta = 1$.

5.3 DATA

Our dataset consists of 525 phone calls to an English-speaking medical emergency service. The call audio is mono-channel, PCM-encoded and sampled at 8000 Hz. The duration of the calls has the mean of 166 s (st. dev. 65 s, IQR 52 s). All calls are manually annotated for questions by trained native English speakers. Each question is annotated with its start and stop time and assigned with one of 13 predefined question labels or an additional label for any question that falls outside of the 13 categories. Figure 6 illustrates these annotations. We observe an initial inter-annotator agreement of $\alpha = 0.8$ [155]. Each

call has been additionally corrected at least once by a different annotator to improve the quality of the data. On average it took roughly 30 minutes to annotate a single call. For our experiments, we choose the five most frequent questions classes, which are explained in Table 4. Out of 24 hours of calls, the questions alone account for only 30 minutes (roughly 2%) of audio. For the experiments we use 5-fold cross-validation stratified by the number of questions in each call, such that calls of different lengths and contents are included in all folds.

We test our model on an additional speech sequence labeling challenge: tracking mentions of medical symptoms in incoming audio. By using another task we gauge the robustness of MultiQT as a general sequence labeling model and not only a question tracker, since symptom utterances in speech carry inherently different linguistic features than questions. As our question-tracking data was not manually labeled for symptoms, we created silver-standard training and test sets automatically by propagating a list of textual keywords from the ground truth human transcripts back onto the audio signal as time stamps with a rule-based algorithm. The initial list contained over 40 medical symptoms, but in the experiment we retain the most frequent five: state of consciousness, breathing, pain, trauma, and hemorrhage.

The utterances that we track are complex phrases with a high variance: There are many different ways to express a question or a medical symptom in conversation. This linguistic complexity sets our research apart from most work in speech labeling which is much closer to exact pattern matching [235].

5.4 EXPERIMENTS

5.4.1 *Setup*

INPUTS. The audio modality is encoded using 40 log-mel features computed with a window of 0.02 s and stride 0.01 s. The textual modality is formed by application of an ASR system to the audio modality. In all reported experiments, only ASR outputs are used and never human transcriptions, both in training and evaluation. The audio input to the ASR is encoded in the same way as described above. The ASR available to us has a purely convolutional architecture similar to the one in [60] with an overall stride of 2. For MultiQT, this means that $T_a = 2T_s$. The ASR is trained on 600 hours of phone calls to medical emergency services in English from the same emergency service provider as the question and symptoms tracking datasets. Both of these are contained in the ASR test set. The ASR is trained using the connectionist temporal classification (CTC) loss function [93] and has a character error rate of 14 % and a word error rate of 31 %. Its feature

dimension is 29 which corresponds to the English alphabet including apostrophe, space and a blank token for the CTC loss.

SYSTEMS. The basic version of MultiQT uses a single softmax cross-entropy loss function and forms a time-bound multimodal representation by concatenating the unimodal representations. We then augment this model in three ways:

1. MultiQT-TF: tensor fusion instead of concatenation following Zadeh et al. [281],
2. MultiQT-MT: auxiliary binary classification with $\beta = 0.5$,
3. MultiQT-TF-MT: combination of 1 and 2.

BASELINES. MultiQT can easily be adapted to a single modality by excluding the respective convolutional transformation f_a or f_s. For example, MultiQT can be trained unimodally on audio by removing f_s and then defining $\mathbf{z}_m^{(t)} = \mathbf{z}_a^{(t)}$ instead of concatenation or tensor fusion. We baseline the multimodal MultiQT models against versions trained unimodally on audio and text. We also compare MultiQT to two distinct baseline models:

1. Random forest (RF)
2. Fully connected neural network (FNN)

Contrary to MultiQT, the baselines are trained to classify an input sequence into a single categorical distribution over the labels. At training, the models are presented with short segments of call transcripts in which all timesteps share the same label such that a single prediction can be made. The baselines are trained exclusively on text and both models represent the windowed transcript as a TF-IDF-normalized bag of words similar to Zhang, Zhao, and LeCun [285]. The bag of words uses word uni- and bigrams, and character tri-, four- and five-grams with 500 of each selected by χ^2-scoring between labels and transcripts on the training set.

HYPERPARAMETERS. We use 1D convolutions for f_a and f_s. For f_a we use three layers with kernel sizes of 10, 20 and 40, filters of 64, 128 and 128 units and strides of 2, 2 and 2 in the first, second and third layer, respectively. For f_s we use two layers with kernel sizes of 20 and 40, filters of 128 and 128 units and strides of 2 and 2. Before each nonlinear transformation in both f_a and f_s we use batch normalization [118] with momentum 0.99 and trainable scale and bias, and we apply dropout [253] with a dropout rate of 0.2. For g we use three fully connected layers of 256 units each and before each nonlinear transformation we use batch normalization as above and apply dropout with a dropout rate of 0.4. We l_2 regularize all learnable parameters with a weighting of 0.1.

		INSTANCE			TIMESTEP		
Model	Modality	P	R	F1	P	R	F1
RF-BOW	T	61.8±3.5	88.5±0.9	72.2±2.2	39.3±1.1	70.4±1.0	48.1±1.0
FNN-BOW	T	42.2±1.4	92.8±0.6	57.5±1.3	38.1±0.7	71.0±1.7	46.9±0.8
MultiQT	A	87.4±1.9	60.6±4.0	70.3±3.1	79.2±1.3	57.8±3.3	65.0±2.4
MultiQT	T	84.2±1.6	78.5±2.8	81.1±2.0	78.8±1.2	69.4±2.0	73.5±1.3
MultiQT	A+T	83.6±2.2	83.3±2.5	**83.3±1.6**	75.7±2.2	73.8±2.3	**74.5±1.3**
MultiQT-MT	A	84.6±5.1	57.4±3.9	66.2±2.9	77.7±5.6	56.0±2.8	62.8±2.0
MultiQT-MT	T	81.9±1.1	80.6±2.8	81.0±1.8	75.9±1.5	71.2±2.4	73.3±1.7
MultiQT-MT	A+T	85.2±2.7	83.2±1.2	**84.1±2.0**	78.5±2.5	74.0±0.7	**76.0±1.1**
MultiQT-TF	A+T	85.0±1.8	83.3±2.6	**83.9±1.7**	78.9±2.1	75.2±2.3	**76.7±1.2**
MultiQT-TF-MT	A+T	85.1±3.2	83.1±1.6	83.8±1.7	78.7±3.7	75.0±1.6	76.5±1.4

Table 5: Question tracking results on audio (A) and text (T) modalities with variations of MultiQT using modality concatenation (MultiQT) or tensor fusion (MultiQT-TF) and the auxiliary task (MultiQT-MT). The evaluation metrics are precision (P), recall (R), and (F1) at the macro level per TIMESTEP or INSTANCE. We report means and standard deviations for five-fold cross-validation runs. All F1 differences are statistically significant at $p < 0.001$, save for between MulitQT [T] & MulitQT-MT [T], and MulitQT [A+T] & MulitQT-TF-MT [A+T] ($p \approx 0.64$). We employ the approximate randomization test with $R = 1000$ and Bonferonni correction [74]. Bold face indicates the highest F1 score within each metric and MultiQT model group.

The FNN model uses the same classifier as is used for g in MultiQT with a dropout rate of 0.3 and an l_2 regularization factor of 0.05.

All neural models are trained with the Adam optimizer [148] using a learning rate of 1×10^{-4}, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and batch size 6 except for those with tensor fusion which use a batch size of 1 due to memory constraints. Larger batch sizes were prohibitive since we use entire calls as single examples but results were generally consistent across different batch sizes. All hyperparameters were tuned manually and heuristically. It takes approximately one hour to train the base MultiQT model on one NVIDIA GeForce GTX 1080 Ti GPU card.

EVALUATION. For each model we report two F1 scores with respective precisions and recalls macro-averaged over the classes.

- TIMESTEP: For each timestep, the model prediction is compared to the gold label. The metrics are computed per timestep and micro-averaged over the examples. This metric captures the model performance in finding and correctly classifying entire audio segments that represent questions and is sensitive to any misalignment.
- INSTANCE: A more forgiving metric which captures if sequences of the same label are found and correctly classified with accep-

tance of misalignment. Here, the prediction counts as correct if there are at least five consecutive correctly labeled time steps within the sequence, as a heuristic to avoid ambiguity between classes. This metric also excludes the non-question label.

The baseline models are evaluated per TIMESTEP by labeling segments from the test set in a sliding window fashion. The size of the window varies from 3 to 9 seconds to encompass all possible lengths of a question with the stride set to one word. Defining the stride in terms of words is possible because the ASR produces timestamps for the resulting transcript per word.

5.4.2 *Results*

LABELING ACCURACY. The results are presented in Table 5. They show that for any model variation, the best performance is achieved when using both audio and text. The model performs the worst when using only audio which we hypothesize to be due to the increased difficulty of the task: While speech intonation may be a significant feature for detecting questions in general, discerning between specific questions is easier with access to transcribed keywords.

Including the auxiliary binary classification task (MultiQT-MT) shows no significant improvement over MultiQT. We hypothesize that this may be due to training on a subset of all questions such that there are unlabelled questions in the training data which add noise to the binary task.

Applying tensor fusion instead of concatenating the unimodal representations also does not yield significant improvements to MultiQT contrary to the findings by Zadeh et al. [281]. Since tensor-fusion subsumes the concatenated unimodal representations by definition and appends all element-wise products, we must conclude that the multimodal interactions represented by the element-wise products either already exist in the unimodal representations, by correlation, are easily learnable from them or are too difficult to learn for MultiQT. We believe that the interactions are most likely to be easily learnable from the unimodal representations.

Comparing any MultiQT variant with INSTANCE and TIMESTEP F1 clearly shows that INSTANCE is more forgiving, with models generally achieving higher values in this metric. The difference in performance between different combinations of the modalities is generally higher when measured per INSTANCE as compared to per TIMESTEP.

The RF and FNN baseline models clearly underperform compared to MultiQT. It should be noted that both RF and FNN achieve F1-scores of around 85 when evaluated per input utterance, corresponding to the input they receive during training. On this metric, FNN also outperforms RF. However, both models suffer significantly from the discrepancy between the training and streaming settings as mea-

sured per the INSTANCE and TIMESTEP metrics; this effect is largest for the FNN model.

REAL-TIME TRACKING. One important use case of MultiQT is real-time labelling of streamed audio sequences and associated transcripts. For this reason, MultiQT must be able to process a piece of audio in a shorter time than that spanned by the audio itself. For instance, given a 1 s chunk of audio, MultiQT must process this in less than 1 s in order to maintain a constant latency from the time that the audio is ready to be processed to when it has been processed. To assess the real-time capability of MultiQT, we test it on an average emergency call using an NVIDIA GTX 1080 Ti GPU card. In our data, the average duration of an emergency call is 166 s.

To simulate real-time streaming, we first process the call in 166 distinct one-second chunks using 166 sequential forward passes. This benchmark includes all overhead, such as the PCIe transfer of data to and from the GPU for each of the forward passes. The choice of 1 s chunk duration matches our production setting but is otherwise arbitrary with smaller chunks giving lower latency and larger chunks giving less computational overhead. In this streaming setting, the 166 s of audio are processed in 1.03 s yielding a real-time factor of approximately 161 with a processing time of 6.2 ms per 1 s of audio. This satisfies the real-time constraint by a comfortable margin, theoretically leaving room for up to 161 parallel audio streams to be processed on the same GPU before the real-time constraint is violated.

When a single model serves multiple ongoing calls in parallel, we can batch the incoming audio chunks. Batching further increases the real-time factor and enables a larger number of ongoing calls to be processed in parallel on a single GPU. This efficiency gain comes at the cost of additional, but still constant, latency since we must wait for a batch of chunks to form. For any call, the expected additional latency is half the chunk duration. We perform the same experiment as above but with different batch sizes. We maintain super real-time processing for batches of up 256 one-second chunks, almost doubling the number of calls that can be handled by a single model.

In the offline setting, for instance for on-demand processing of historical recordings, an entire call can be processed in one forward pass. Here, MultiQT can process a single average call of 166 s in 10.9 ms yielding an offline real-time factor of 15,000. Although batched processing in this setting requires padding, batches can be constructed with calls of similar length to reduce the relative amount of padding and achieve higher efficiency yet.

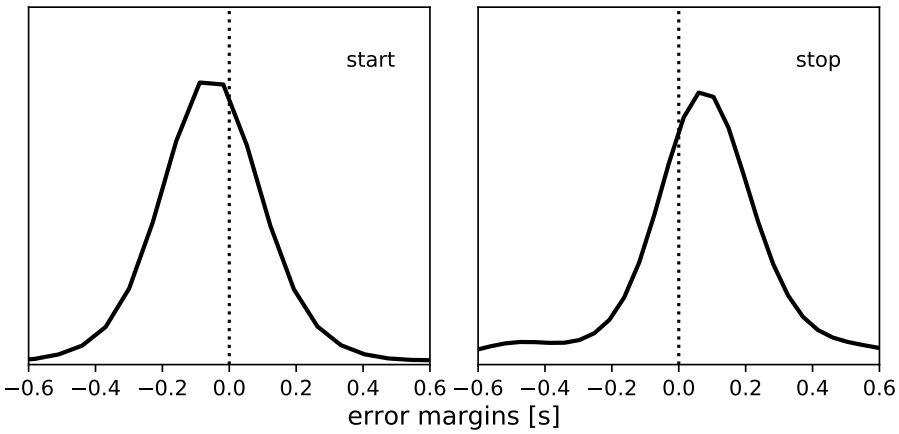


Figure 8: Error margin distributions for start and stop timestamps of question sequences. The dotted lines depict the ground truth start and stop timestamps.

5.5 DISCUSSION

LABEL CONFUSION. We analyze the label confusion of the basic MultiQT model using both modalities on the TIMESTEP metric. Less than 1% of all incorrect timestamps correspond to question-to-question confusions while the two primary sources of confusion are incorrect labelings of 1) "None" class for a question and 2) of a question with the "None" class. The single highest confusion is between the "None" class and "Q4" which is the least frequent question. Here the model has a tendency to both over-predict and miss: ca 40% of predicted "Q4" are labeled as "None" and 40% of "Q4" are predicted as "None". In summary, when our model makes an error, it will most likely 1) falsely predict a non-question to be a question or 2) falsely predict a question to be a non-question; once it discovers a question, it is much less likely to assign it the wrong label.

MODEL DISAGREEMENT. We examined the inter-model agreement between MultiQT trained on the different modes. The highest agreement of ~90% is achieved between the unimodal text and the multimodal models whereas the lowest agreement was generally between the unimodal audio and any other model at ~80%. The lower agreement with the unimodal audio model can be attributed to the generally slightly lower performance of this model compared to the other models as per Table 5.

QUESTION MARGINS. In Figure 8, we visualize the distribution of the errors made by the model per TIMESTEP. For each question regarded as matching according to the INSTANCE metric we compute the number of seconds by which the model mismatched the label sequence on the left and right side of the label sequence, respectively.

	Permuted		INSTANCE			TIMESTEP		
Modality	Training	Test	P	R	F1	P	R	F1
A+T	Yes	T	82.2±4.9	60.1±5.6	68.6±5.7	79.0±4.7	58.4±3.7	64.7±3.5
A+T	Yes	A	82.6±3.2	75.9±2.9	78.7±1.6	78.3±2.4	68.3±2.7	72.3±1.1
A+T	Yes	-	86.3±1.6	83.8±2.8	84.8±2.0	80.4±1.0	74.1±2.2	76.9±1.3
A+T	No	T	0.0±0.0	0.0±0.0	0.0±0.0	16.2±0.0	16.7±0.0	16.4±0.0
A+T	No	A	89.5±3.1	69.2±4.4	77.0±2.5	84.3±2.6	63.7±3.5	71.0±2.0
A+T	No	-	83.6±2.2	83.3±2.5	83.3±1.6	75.7±2.2	73.8±2.3	74.5±1.3
A	No	-	87.4±1.9	60.6±4.0	70.3±3.1	79.2±1.3	57.8±3.3	65.0±2.4
T	No	-	84.2±1.6	78.5±2.8	81.1±2.0	78.8±1.2	69.4±2.0	73.5±1.3

Table 6: Results from the modality ablation on the MultiQT model. We compare multimodal MultiQT trained with the audio (A) and text (T) modalities temporally permuted in turn during training with probability $p_a = 0.1$ and $p_s = 0.5$ to MultiQT trained without modality permutation, unimodally and multimodally (some results copied from Table 5). We can obtain robustness to loosing a modality while maintaining (or even slightly improving) the multimodal performance. All results are based on five-fold cross-validation as in Table 5.

We see that the model errors are normally distributed around a center value that is shifted towards the outside of the question by slightly less than 100 ms. The practical consequence is that the model tends to make predictions on the *safe side* by extending question segments slightly into the outside of the question.

MODALITY ABLATION. To evaluate the model's robustness to noise in the modalities, we remove all information from one of the modalities in turn and report the results in Table 6. We remove the information in a modality by randomly permuting the entire temporal axis. This way we retain the numerical properties of the signal which is not the case when replacing a modality by zeros or noise. To increase MultiQT's robustness to this modality ablation, we apply it at training so that for each batch example we permute the temporal axis of the audio or text modality with some probability p_a or p_s. We choose $p_a = 0.1$ and $p_s = 0.5$ since the model more easily develops an over-reliance on the text-modality supposedly due to higher signal-to-noise ratio. The results are listed in Table 6 along with results for MultiQT from Table 5 for easy reference. We observe that the basic MultiQT model suffers significantly from permutation of the text modality and less so for audio which suggests that it relies on the audio only for supportive features. Training MultiQT with the random temporal permutation forces learning of robustness to loosing all information in a modality. We see that the results when removing a modality almost reach the level achieved when training

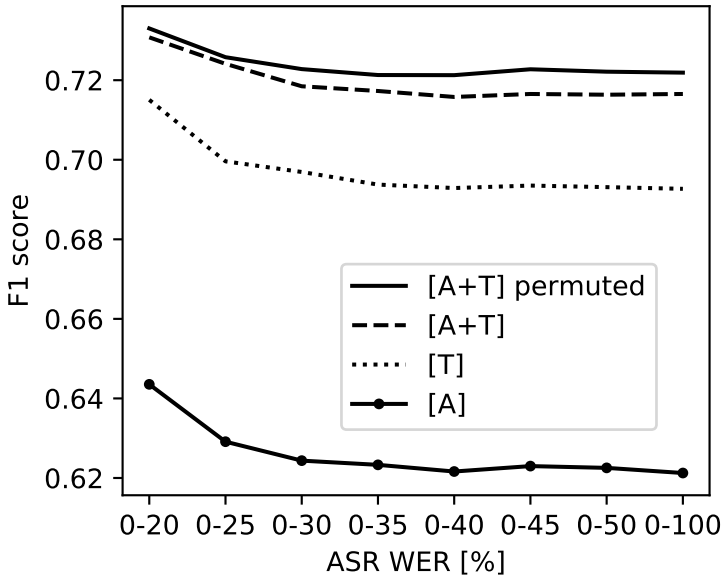


Figure 9: Relation between TIMESTEP F1 and WER on call-taker utterances without the "None" label.

exclusively on that modality while still maintaining the same (or better) performance of the basic MultiQT model.

RELATION TO ASR. In Figure 9, we plot the performance of the multimodal model on different subsets of the test split by the maximum WER of the ASR (measured only on the call-taker utterances). This evaluation compares the micro-averaged model F1-score when increasing the noise on the textual input. We see that regardless of the modality, the performance is the highest for calls with very low WER. We observe that the performance improvement of using both modalities over unimodal text or unimodal audio increases as we include noisy samples. This implies that multi modality increases robustness. Training on permuted inputs additionally improves the performance on noisy data.

The evaluation of MultiQT in our paper has thus far been only in relation to one particular ASR model with CTC loss [93], where our system displays significant gains from multimodal learning. Yet, do these results hold with another ASR system, and in particular, are the multimodal gains still significant if WER decreases and produced text quality increases? For an initial probing of these questions, we replace the fully convolutional ASR with a densely-connected recurrent architecture with convolutional heads. This model is similar to the one in [2] but also uses dense bottleneck layers. With this model the transcription quality improves by around +4% in WER, while the F1-scores of MultiQT still strongly favor the multimodal approach, by +6.15 points absolute over text-only. We argue that in a real-world scenario with high WER and limited in-domain training data, the gains

warrant learning from joining the text and audio views on the input speech when learning a question tracker. Alternatively, the ASR model itself could be extended into a multitask learning setup to jointly track questions and transcribe speech; we defer that line of work for future research. On a practical note, for this multitask approach, the data must be fully transcribed by human annotators in addition to the question annotatations. This is generally more time consuming and expensive than exclusively annotating questions.

QUALITATIVE ANALYSIS. We analyze the model predictions on a subset of 21 calls to identify the most likely reasons for incorrect labeling. We find that in over half of the analysed cases the incorrect prediction is triggered either by a question-related keyword uttered in a non-question sentence or by a question asked in the background by a caller that was not assigned a label. We also encounter undetected questions that have a very noisy ASR transcript or are asked in an unusual way.

SYMPTOM LABELING. The experiment with our silver-standard symptoms data shows a trend that is similar to question tracking: The dual-modality MultiQT scores an INSTANCE F1 score of 76.9 for a +1.8 absolute improvement over the best single modality. Text-only is the runner up (-1.8 F1) while audio-only lags behind with a significant -23.6 decrease in F1. At the same time, a simple text-only keyword matching baseline scores at 73.7. We argue that symptom tracking strongly favors text over audio because the distinctive audio features of questions, such as changes in intonation, are not present when communicating symptoms in speech.

5.6 RELATED WORK

The broader context of our work is to track the dialogue state in calls to emergency medical services, where conversations are typically formed as sequences of questions and answers that pertain to various medical symptoms. The predominant approach to dialogue state tracking (DST) in speech is to first transcribe the speech by using ASR [104, 105, 201]. In our specific context, to entirely rely on ASR is prohibitive because of significantly higher WER in comparison to standard datasets. To exemplify, while WER is normally distributed with a mean of 37.6% in our data, the noisiest DST challenge datasets rarely involve with WER above 30% [122] while standard ASR benchmarks offer even lower WER [214]. None of the standard ASR scenarios thus directly apply to a real-life ASR noise scenario.

From another viewpoint, work in audio recognition mainly involves with detecting simple single-word commands or keyword spotting [3], recognizing acoustic events such as environmental or urban sounds [221,

236, 279] or music patterns, or document-level classification of entire audio sequences [177]. McMahan and Rao [190] provide a more extensive overview. While approaches in this line of work relate to ours, e.g. in the use of convolutional networks over audio [234, 235], our challenge features questions as linguistic units of significantly greater complexity.

Finally, research into multimodal or multi-view deep learning [171, 203] offers insights to effectively combine multiple data modalities or views on the same learning problem. However, most work does not directly apply to our problem: i) the audio-text modality is significantly under-represented, ii) the models are typically not required to work online, and iii) most tasks are cast as document-level classification and not sequence labeling [282].

5.7 CONCLUSIONS

We proposed a novel approach to speech sequence labeling by learning a multimodal representation from the temporal binding of the audio signal and its automatic transcription. This way we learn a model to identify questions in real time with a high accuracy while trained on a small annotated dataset. We show the multimodal representation to be more accurate and more robust to noise than the unimodal approaches. Our findings generalize to a medical symptoms labeling task, suggesting that our model is applicable as a general-purpose speech tagger wherever the speech modality is coupled in real time to ASR output.

ACKNOWLEDGEMENTS

The authors are grateful to the anonymous reviewers and area chairs for the incisive and thoughtful treatment of our work.

Part III

UNSUPERVISED LEARNING

6

AN OVERVIEW OF UNSUPERVISED NEURAL SPEECH REPRESENTATION LEARNING

ABSTRACT

Unsupervised representation learning for speech processing has matured greatly in the last few years. Work in computer vision and natural language processing has paved the way, but speech data offers unique challenges. As a result, methods from other domains rarely translate directly. We review the development of unsupervised representation learning for speech over the last decade. We identify two primary model categories: self-supervised methods and probabilistic latent variable models. We provide a comprehensive model taxonomy and describe evaluation procedures for speech representation learning. Finally, we discuss and compare models from the two categories.

6.1 INTRODUCTION

Representation learning has shaped modern computer vision [247] and natural language processing [68], and more recently speech processing has been subject to the same development [13]. Representation learning has been defined as *"learning representations of the data that make it easier to extract useful information when building classifiers or other predictors"* [22]. Unsupervised representation learning is concerned with learning useful representations without the use of human annotations. Usually, a model is first pre-trained on a task where plenty of data is available. The model is then fine-tuned, or used to extract input representations for a smaller model, targeting a task with limited training data. In computer vision, both supervised [101, 247, 256] and unsupervised [72, 219] representation learning have gained attention with supervised representation learning driven by the availability of large annotated datasets [65]. For text and speech, pre-training is usually unsupervised as labeled data is difficult to obtain. Although work on text has paved the way, and the two fields share many characteristics, learning representations from speech is a problem faced with a unique set of challenges.

In this paper, we survey work on unsupervised representation learning for speech processing from within the last decade. From a methodological perspective, we identify two primary model categories, namely models based on self-supervised learning and probabilistic latent variable models. We provide a methodological review of the design choices related to each of the model categories and develop a model taxon-

omy that highlights the different directions of work. We then provide a comparison of the methods based on their respective evaluation procedures. Finally, based on the model taxonomy, we discuss the differences between models from the two categories and identify future challenges and directions for research in speech representation learning.

6.2 UNSUPERVISED REPRESENTATION LEARNING

In the following, we group previous work into *self-supervised models* and *probabilistic latent variable models*, and take a *model* to comprise a neural architecture and a corresponding learning algorithm. A schematic overview is found in figure 10. These categories are neither exhaustive nor mutually exclusive, but allow us to focus on the characteristics that have shaped different branches of research.

With emphasis on the recent successes in the field, we aim to cover literature from within the last 10 years. While a complete description of all relevant models is not within the scope of this work, we sketch important technicalities when they are particularly descriptive of a certain class of models. We start out by defining our high-level notation and some conventions to ease discussion.

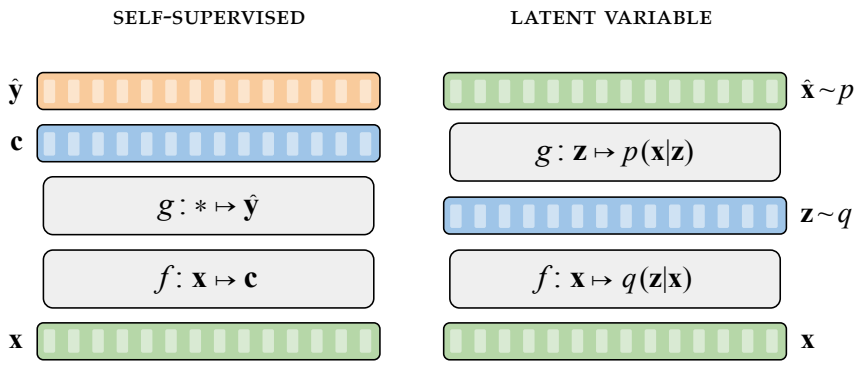


Figure 10: A schematic overview of the two groups of models covered in this survey. **Left**: A model trained with self-supervised learning. There is no strict specification of how to configure a self-supervised model. In this review, we take these models to consist of two functions $f(\cdot)$ and $g(\cdot)$. After pre-training, $f(\cdot)$ is either fine-tuned or used for extracting features $\mathbf{c}$ for a downstream task. See figure 11. $g(\cdot)$ is an auxiliary function used to accommodate the self-supervised pre-training task. Thus, it might operate on $\mathbf{x}$, $\mathbf{c}$ or another hidden layer (as indicated by *) to approximate some data induced target $\mathbf{y}$. Typically, the parameter space of $g(\cdot)$ will be much smaller than that of $f(\cdot)$. **Right**: A probabilistic latent variable model. In contrast to the self-supervised model, the functions $f(\cdot)$ and $g(\cdot)$ are used to learn the parameters of the distributions q and p. Here, the latent variable $\mathbf{z}$ is commonly used for representation learning.

6.2.1 *Notation*

We use the subscript $i{:}j$ with $i \leqslant j$ to denote a vector sequence $\mathbf{a}_{i:j}$ containing elements $\mathbf{a}_i$ through $\mathbf{a}_j$. The absence of a subscript indicates that the vector is not part of any sequence. We denote model input as $\mathbf{x}_{1:T}$ which, in practice, might be either a spectrogram or the raw speech signal, but we do not distinguish between the two in notation as it is not essential to understand the models. Also, models commonly downsample the temporal dimension, but again, this is not crucial to understand the models, so we maintain a notation based only on a single temporal dimension $t \in \{1, \ldots, T\}$.

When discussing self-supervised models, we use $\mathbf{c}_{1:T}$ to denote a contextualized representation. For stochastic latent variable models we use $\mathbf{z}_{1:T}$ as is customary to the field. The representation at a single time-step, $\mathbf{c}_t$ or $\mathbf{z}_t$, is typically a function of a larger part of the input space than just $\mathbf{x}_t$, in some cases the entire input sequence $\mathbf{x}_{1:T}$.

While some models are frozen and produce representations used as input for downstream tasks (FRZ, table 7), others are designed to be fine-tuned (FTN, table 7). We illustrate this difference in figure 11. In either case, we use $f(\cdot)$ to denote the model that is used for the downstream task. We use $g(\cdot)$ to denote any auxiliary model components (e.g., for a reconstruction task we might have $g : \mathbf{c}_t \mapsto \hat{\mathbf{x}}_t$). When a model can be naturally subdivided into multiple components, we simply use $f_*(\cdot)$ where $*$ may be any convenient descriptor. Finally, we often use a subscript when defining a loss, $\mathcal{L}_i$, to imply that the total loss is computed as a sum over i.

6.3 SELF-SUPERVISED MODELS

Self-supervised learning is a subset of unsupervised learning [262]. Where other unsupervised learning methods can be seen as a means to an end in itself (e.g., clustering, dimensionality reduction, or data generation), self-supervised learning takes the form of a pretext task that only adds value when associated with a downstream task. This makes self-supervised learning tie naturally with semi-supervised learning, but it may also be part of a fully unsupervised setup [10]. Self-supervised learning is often characterized by automatically deriving the target from the input or other unlabeled examples [211].

6.3.1 *Predictive models*

Similar to classic autoregressive language models [193], contextualized speech representations can be learned by predicting future values of a simple representation [50, 52, 130, 206, 241] (PRD, table 7). Modeling spectrograms directly, autoregressive predictive coding (APC)

[52] is perhaps the simplest example in this category. The forward pass and loss are computed as

$$\mathbf{c}_t = f(\mathbf{x}_{1:t}) \tag{36}$$

$$\hat{\mathbf{x}}_{t+k} = g(\mathbf{c}_t) \tag{37}$$

$$\mathcal{L}_t = \|\hat{\mathbf{x}}_{t+k} - \mathbf{x}_{t+k}\|_1 \ . \tag{38}$$

Here, $f(\cdot)$ and $g(\cdot)$ are parameterized by neural networks such that each $\mathbf{c}_t$ is only conditioned on previous inputs $\mathbf{x}_{1:t}$ and $\hat{\mathbf{x}}_{t+k}$ is computed step-wise. Chung et al. [52] use a stack of unidirectional LSTMs for $f(\cdot)$ and a linear regression layer for $g(\cdot)$. Tasks that seek to predict or reconstruct the input are very common. In the literature, these are often jointly referred to as reconstruction tasks (REC, table 7) [175, 270], although this is somewhat misleading in the case of prediction.

Contrary to a text-based autoregressive language model, the APC model is not restricted to next-step prediction. Instead, it predicts $k > 0$ steps ahead in order to ensure that the model does not learn a trivial solution by exploiting the smoothness of the signal. Depending on the downstream task, we are often interested in learning so-called slow features that will typically span multiple input frames [274]. Even the smallest linguistic units of speech, phonemes, tend to span 0.1 seconds on average [87], whereas spectrogram frames $\mathbf{x}_t$ are typically computed at 0.01 second intervals.

However, sometimes local smoothness is explicitly used [8, 127, 128]. In early work, Badino et al. [8] propose a step-wise bottleneck

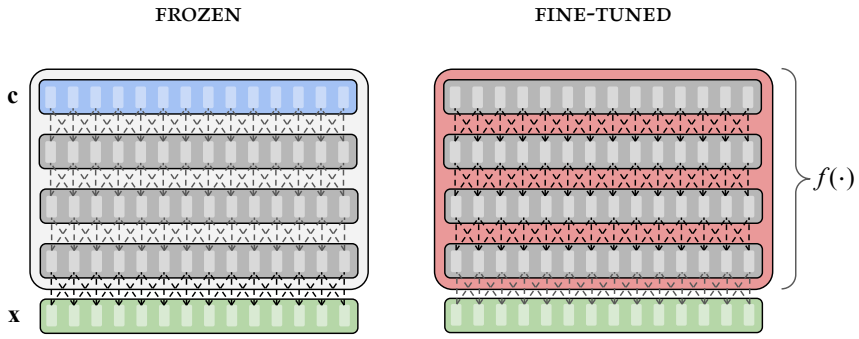


Figure 11: A simple visualization of the two most common approaches to representation learning. **Left**: A frozen (FRZ, table 7) pre-trained model $f(\cdot)$ is used to extract features from the input sequence $\mathbf{x}_{1:T}$. The learned local representations $\mathbf{c}_{1:T}$ are used as input to a model which is trained for a downstream task. In this visualization, the representations are extracted from the top layer of the model, but in practice any intermediate layer could potentially be used. **Right**: A pre-trained model $f(\cdot)$ is fine-tuned (FTN, table 7) for a downstream task. Depending on the fine-tuning strategy, all or a subset of the parameter space is adjusted. Usually, it is necessary to add additional parameters (e.g., an output layer) that must be trained from scratch to fit the pre-trained model to a new task.

model that learns to predict $\mathbf{x}_{t+1}$ from $\mathbf{x}_t$. Since phonetic information changes slowly, $\mathbf{x}_{t+1}$ is argued to be a noisy version of $\mathbf{x}_t$ encoding almost the same phonetic content. Thus, the model is akin to a denoising autoencoder.

Next-step prediction has also been frequently used for generative models (GEN, table 7). The popular speech synthesis model WaveNet is perhaps the most obvious example [207]. Next-step prediction has also been used to infer phoneme boundaries in an unsupervised manner [192]. While this work does not consider representation learning, the models might learn useful features similar to those learned with APC.

6.3.2 *Contrastive models*

Speech contains localized noise (e.g., phase shifts) that does not inform slow feature learning. Thus, directly modeling speech might not be the best way to learn contextualized representations. Contrastive predictive coding (CPC) [206] targets a local variable $\mathbf{v}_{1:T}$, learned from the model input $\mathbf{x}_{1:T}$, instead of the input itself. The forward pass is

$$\mathbf{v}_t = f_v(\mathbf{x}_{t-r:t+r}) \tag{39}$$

$$\mathbf{c}_t = f_c(\mathbf{v}_{1:t}) \tag{40}$$

$$\hat{\mathbf{v}}_{t,k} = g_k(\mathbf{c}_t) \ , \tag{41}$$

where $f_v(\cdot)$ is a convolutional neural network, such that each $\mathbf{v}_t$ only encodes information from a limited receptive field $2r+1$. Again, $f_c(\cdot)$ should be limited to condition each $\mathbf{c}_t$ on previous time-steps $\mathbf{v}_{1:t}$ and $g_k(\cdot)$ is a step-wise transformation (e.g., a linear regression layer). The loss is based on noise constrastive estimation [95] and is given by

$$\mathcal{L}_{t,k} = -\log\left(\frac{\exp(\hat{\mathbf{v}}_{t,k}^T \mathbf{v}_{t+k})}{\sum_{n\sim\mathcal{D}} \exp(\hat{\mathbf{v}}_{t,k}^T \mathbf{v}_n)}\right) \ . \tag{42}$$

Here, $\mathcal{D}$ is a set of indices including the target index $t+k$ and negative samples drawn from a proposal distribution, which is typically taken to be a uniform distribution over the set $\{1,\dots,T\}$. Note that the loss is also indexed by k to show that CPC targets multiple offsets. The APC model is easily extended in a similar way [51].

Crucially, we cannot simply predict $\mathbf{v}_{t+k}$ from $\mathbf{c}_t$ with an ℓ_1 loss. This would cause $f_v(\cdot)$ to collapse to a trivial solution, such as setting all $\mathbf{v}_t$ equal. With a contrastive loss on the other hand, setting all $\mathbf{v}_t$ equal would cause $\mathcal{L}_{k,t}$ to be constant at a value no better than a random baseline.

A model closely related to the original CPC model is wav2vec [241]. It uses a different parameterization of the functions $f_v(\cdot)$ and $f_c(\cdot)$,

and modifies the loss to consider a binary prediction task, such that we have

$$\mathcal{L}_{t,k} = -\log(\sigma(\hat{\mathbf{v}}_{t,k}^{T}\mathbf{v}_{t+k})) - \sum_{n\sim\mathcal{D}} \log(\sigma(-\hat{\mathbf{v}}_{t,k}^{T}\mathbf{v}_{n})) \ . \tag{43}$$

This model was among the first to show that learned representations can be used to improve end-to-end speech recognition. As we will see, the wav2vec framework has gradually evolved to shape state-of-the-art representation learning for speech.

6.3.3 *Masking-based models*

One downside of predictive tasks is that models are primarily unidirectional. Some work has extended APC and CPC inspired models with separate encoders operating in opposite directions [32, 141, 173], but these models are still restricted to process left and right context separately. Inspired by the masked language model task used for text-based representation learning [68], several papers have used masking to overcome this challenge (MSK, table 7). Masking refers to replacing parts of the input with zeros or a learned masking vector. For zero-masking [42, 129, 172, 176, 273], we have

$$\mathbf{c}_t = f(\mathbf{x}_{1:T} \circ \mathbf{m}_{1:T}) \tag{44}$$

$$\hat{\mathbf{x}}_t = g(\mathbf{c}_t) \tag{45}$$

$$\mathcal{L}_t = \|\hat{\mathbf{x}}_t - \mathbf{x}_t\|_1 \ , \tag{46}$$

where the $\circ$ operator denotes the Hadamard product, $f(\cdot)$ is typically a transformer encoder or a bidirectional recurrent neural network, $g(\cdot)$ is a step-wise transformation, and $\mathbf{m}_{1:T}$ is a mask such that $m_{t,i} \in \{0, 1\}$. Alternatively, $\mathbf{m}_{1:T}$ is used to select which $\mathbf{x}_t$ are replaced by a learned masking vector. The entries of $\mathbf{m}_{1:T}$ are determined by some stochastic policy. One frequent inspiration is SpecAugment [214], which was originally proposed for supervised speech recognition and applies frequency and time masking to spectrogram representations. While temporal masking is most common, frequency masking has also been adopted for representation learning [273]. A simple, yet popular, masking strategy is to draw a proportion of input indices $t_i \sim \{1, \ldots, T - M\}$ without replacement, and then mask $\{t_i, \ldots, t_i + M\}$ [13, 111, 172].

Combining masking with a contrastive loss, wav2vec 2.0 was the first work to show that a competitive speech recognition model can be learned by fine-tuning a pre-trained model with as little as 10 minutes of labeled data. For this model

$$\mathbf{v}_t = f_v(\mathbf{x}_{t-r:t+r}) \tag{47}$$

$$\mathbf{c}_t = f_c(\mathbf{v}_{1:T} \circ \mathbf{m}_{1:T}) \tag{48}$$

$$\mathbf{q}_t = g_q(\mathbf{v}_t) \ . \tag{49}$$

Here, $f_v(\cdot)$ is a convolutional neural network, $f_c(\cdot)$ is a transformer encoder [266] and $g_q(\cdot)$ is a quantization module used to learn targets from the localized variable $\mathbf{v}_{1:T}$. Computing quantized targets this way requires an extra loss term, which we will present when we discuss quantization in general below. The contrastive loss for wav2vec 2.0 is similar to that of the CPC model,

$$\mathcal{L}_t = -\log\left(\frac{\exp(S_c(\mathbf{c}_t, \mathbf{q}_t))}{\sum_{n\sim\mathcal{D}}\exp(S_c(\mathbf{c}_t, \mathbf{q}_n))}\right) \quad , \tag{50}$$

where $S_c(\cdot)$ is the cosine similarity and the negative samples in $\mathcal{D}$ are sampled from other masked time-steps.

In general, masking is less data efficient than prediction, as only the masked portion of the input is non-trivial to reconstruct. For this reason, the loss might be computed as

$$\mathcal{L}_t = \|(\hat{\mathbf{x}}_t - \mathbf{x}_t) \circ (\mathbf{1} - \mathbf{m}_t)\|_1 \quad . \tag{51}$$

Non-autoregressive predictive coding (NPC) [174] tries to resolve this by using a convolutional neural network where the kernel is masked instead of the input. That is, the model is prevented from using $\mathbf{x}_t$ for its corresponding reconstruction $\hat{\mathbf{x}}_t$. This allows for complete data utilization, but limits the amount of context encoded in the learned representation. Figure 12 summarizes the models discussed so far.

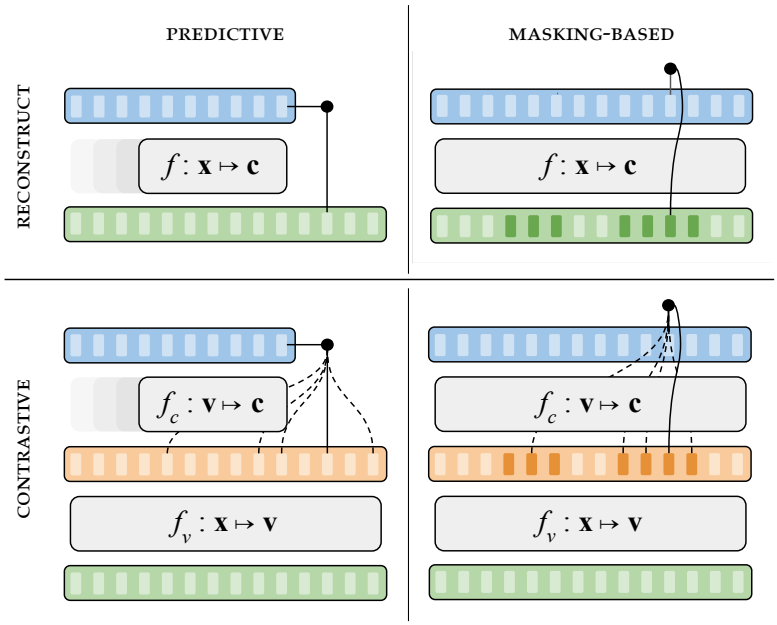


Figure 12: Schematic overview of self-supervised approaches. The subscript of the temporal dimension has been left out for notational simplicity. Each subfigure illustrates the loss computation for a single time-step. Examples of models in each category include APC (PRD + REC), Mockingjay (MSK + REC), CPC (PRD + CON) and wav2vec 2.0 (MSK + CON).

6.3.4 *Quantization*

Several models enforce a discrete latent space by quantizing the vector representation (QTZ, table 7). A popular approach for online vector quantization is to use the Gumbel softmax [123, 186]. This approach corresponds to approximate differentiable sampling from a categorical distribution. Another common approach employs a codebook of learnable representations. This approach was popularized with the VQ-VAE [265], and thus, we refer to it as the VQ-VAE approach. Both approaches employ non-differentiable operations and obtain gradients using the straight-through estimator [23].

6.3.4.1 *Gumbel-softmax approach*

Say we want to quantize a vector $\mathbf{v}$ such that it takes one of K possible values. We first map it to $\mathbf{l} \in \mathbb{R}^K$ and then map $\mathbf{l}$ to a probability vector $\mathbf{p} \in \mathbb{R}^K$ via the Gumbel softmax given by

$$p_i = \frac{\exp(l_i + n_i)/\tau}{\sum_k^K \exp(l_k + n_k)/\tau} \tag{52}$$

for $i = 1, \ldots, K$, where τ is a temperature parameter and $\mathbf{n} \in \mathbb{R}^K$ is a random vector with $n_i = -\log(-\log(u_i))$ for $u_i \sim \mathcal{U}(0, 1)$. For $\tau \to 0$, $\mathbf{p}$ approaches a one-hot vector. The Gumbel noise $\mathbf{n}$ is a practical way to sample from the untempered categorical distribution (i.e., $\tau = 1$)[1]. In order to ensure that that we end up with a discrete sample, $\mathbf{p}$ is mapped to a one-hot vector using a function $\varphi(\cdot)$, such that $\varphi(\mathbf{p})_i = 1$ if $i = \arg\max_j p_j$ and 0 otherwise. However, this function is clearly non-differentiable, so we have to rely on the straight-through estimator and assume during training that the Jacobian $\partial\, \varphi / \partial\, \mathbf{p}$ equals the identity matrix. The one-hot vector can then be used for a codebook lookup to obtain the final quantized vector (e.g., $\mathbf{q}_t$ in eq. 49).

The wav2vec 2.0 quantization module (eq. 49) relies on the Gumbel softmax approach. To support this, a diversity loss is added to the task specific loss (eq. 50) to ensure even utilization of the codebook vectors

$$\mathcal{L} = -H(\widetilde{\mathbf{p}})\frac{1}{D} \ , \tag{53}$$

where $H(\cdot)$ is the entropy and $\widetilde{\mathbf{p}}$ is the untempered version of $\mathbf{p}$ without Gumbel noise.

1 Assume a categorical random variable taking values $1, \ldots, K$ with probabilities $\pi_1, \ldots, \pi_K$. The underlying distribution can be sampled by drawing $i = \text{argmax}_k(l_k + n_k)$, where the $n_k = -\log(-\log(u_k))$ with $u_k \sim \mathcal{U}(0, 1)$ are Gumbel distributed and $l_k = \log(\tilde{\pi}_k)$ with $\tilde{\pi}_k / \sum_j \tilde{\pi}_j = \pi_i$. The Gumbel softmax replaces the argmax by a softmax.

		Model and task design						Resolution			Usage	
MODEL	PUB. DATE	MSK	PRD	CON	REC	QTZ	GEN	LOC	GLB	VAR	FRZ	FTN
Self-supervised models												
Audio Word2vec [54]	2016 Mar.	✓	✗	✗	✓	✗	✗	✗	✓	✗	✓	✗
Speech2Vec [49]	2018 Mar.	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗
Unspeech [197]	2018 Apr.	✗	✓	✓	✗	✗	✗	✗	✓	✗	✓	✗
CPC [206]	2018 Jul.	✗	✓	✓	✗	✗	✗	✓	✗	✗	✓	✗
PASE [217]	2019 Apr.	✗	✗	✓	✓	✗	✗	✓	✗	✗	✓	✓
APC [52]	2019 Oct.	✗	✓	✗	✓	✗	✗	✓	✗	✗	✓	✗
wav2vec [241]	2019 Apr.	✗	✓	✓	✗	✗	✗	✓	✗	✗	✓	✗
Mockingjay [176]	2019 Oct.	✓	✗	✗	✓	✗	✗	✓	✗	✗	✓	✓
wav2vec 2.0 [13]	2020 Jun.	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓
NPC [174]	2020 Nov.	✓	✗	✗	✓	✓	✗	✓	✗	✗	✓	✗
DeCoAR 2.0 [172]	2020 Dec.	✓	✗	✗	✓	✓	✗	✓	✗	✗	✓	✗
SCPC [25]	2021 Jun.	✗	✓	✓	✗	✗	✗	✓	✗	✓	✓	✗
HuBERT [111]	2021 Jun.	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✓
Probabilistic latent variable models												
VRNN [56]	2015 Jun.	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗
SRNN [85]	2016 May	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗
HMM-VAE [79]	2017 Mar.	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗
ConvVAE [113]	2017 Apr.	✗	✗	✗	✓	✗	✓	✗	✓	✗	✓	✗
FHVAE [114]	2017 Sep.	✗	✗	✗	✓	✗	✓	✓	✓	✗	✓	✗
VQ-VAE [265]	2017 Nov.	✗	✗	✗	✓	✓	✓	✓	✗	✗	✓	✗
BHMM-VAE [89]	2018 Mar.	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗
STCN [1]	2019 Feb.	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗
FDMM [142]	2019 Oct.	✗	✗	✗	✓	✗	✓	✓	✓	✗	✓	✗
ConvDMM [144]	2020 Jun.	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗

Table 7: Selected models classified according to the binary attributes identified throughout the text. Within the two categories, the models are sorted according to publication date of the first version on arXiv which might differ from the year indicated by the citation. We discuss this model taxonomy in section 6.6. MSK means that the model uses masking, PRD that it uses prediction, CON that it uses a contrastive loss, REC that it uses a reconstruction loss, QTZ that it uses quantization, GEN that it is generative, FRZ that frozen representations are extracted for downstream tasks, FTN that the model is fine-tuned for downstream tasks, LOC that the model learns local representations and GLO that it learns a single global representation.

6.3.4.2 *VQ-VAE approach*

Instead of directly parameterizing a probability distribution as in the Gumbel softmax, a vector $\mathbf{v}$ can be quantized by mapping it to the closest codebook vector $\mathbf{e}_k$ with gradients again obtained by the straight-through estimator. Specifically, given a learned codebook $\mathbf{e} \in \mathbb{R}^{K \times D}$, where K is the codebook size and D is the dimensionality of each codebook vector $\mathbf{e}_k$, the quantized representation $\mathbf{q}$ of $\mathbf{v}$ is obtained as,

$$\mathbf{q} = \mathbf{e}_k \ , \quad \text{where} \quad k = \arg\min_j \left\| \mathbf{v} - \mathbf{e}_j \right\|_2 \ . \tag{54}$$

The straight-through estimator is applied to define the gradient of this operation, i.e. the gradient of $\mathbf{v}$ wrt. to the loss is set equal to that of $\mathbf{q}$. Codebook learning is facilitated by a two-term auxiliary loss similar to classical vector quantization dictionary learning [35, 251]. Gradients for the codebook vectors are given solely by a vector quantization term, which moves codebook vectors $\mathbf{e}_k$ closer to the non-quantized vectors $\mathbf{v}$. A so-called commitment term is added to ensure that non-quantized vectors do not grow unboundedly by enforcing the encoder to keep them close to a codebook vector.

$$\mathcal{L} = \underbrace{\|\text{sg}\left[\mathbf{v}\right] - \mathbf{e}\|_2^2}_{\text{vq}} + \underbrace{\beta\|\mathbf{v} - \text{sg}\left[\mathbf{e}\right]\|_2^2}_{\text{commitment}} , \tag{55}$$

where $\text{sg}[\mathbf{x}] = \mathbf{x}$ is the so-called stop-gradient operator with the property that $\frac{d}{dx_i}\text{sg}[\mathbf{x}] \equiv 0$ for all i and β is a scalar hyperparameter.

Although vector quantization was introduced by the VQ-VAE which is, in some respects, a latent variable model, it has been applied extensively to self-supervised methods but with variations between individual approaches. The VQ-CPC model [204] quantizes $\mathbf{v}$ in the original CPC (eq. 39) and augments its loss (eq. 42) by addition of the commitment term in eq. 55. Instead of learning the codebook via the VQ term, it is updated as a moving average of $\mathbf{v}$ as initially suggested in [265]. The vq-wav2vec [12] is defined similarly but adds the full auxiliary loss to the wav2vec loss (eq. 43) in place of moving averages.

6.3.4.3 *Motivation*

Similar to how quantization approaches differ between works, so do the motivations provided for employing them. The vq-wav2vec [9, 12] quantizes $\mathbf{v}_{1:T}$ before feeding them to the context network $f_c(\cdot)$. Here, the motivation is to apply natural language processing models, like BERT [68], to the discrete representations afterwards. Other work use quantization as a bottleneck in order to "*limit model capacity*" [53, 172]. Quantized representations have also been used for speech segmentation [45, 139]. Finally, Chung, Tang, and Glass [53] explore quantization between different layers in the APC model. They find that applying quantization to the output $\mathbf{c}_{1:T}$ yields the best results on a downstream phoneme classification task. However, the continuous representations consistently perform better than their quantized counterparts.

Given our previous discussion of how it might not be beneficial to model localized noise, quantization in wav2vec 2.0 seems well motivated, as it enforces the target representation $\mathbf{q}_{1:T}$ to discard noise. Taking this idea further, the HuBERT model [111] uses offline quantization to learn categorical targets. Initially, spectrogram features are used to learn frame-wise labels with k-means clustering. A model similar to wav2vec 2.0, but without the online quantization, is then

trained to infer labels for masked time-steps. Since quantization is offline, this model does not need to rely on a contrastive loss, but can infer the target class directly. The offline quantization also ensures more stable training, as targets do not change abruptly.

6.3.5 *Global representations*

All self-supervised models covered so far learn representations that maintain a temporal resolution proportional to the input resolution. We say that they learn local representations (LOC, table 7) even though each $\mathbf{c}_t$ might depend on the entire input sequence. In the following, we cover models that output a single global representation (GLB, table 7).

Early work on global speech representation learning takes inspiration from the autoencoder framework [153]. Chung et al. [54] propose a simple sequence-to-sequence autoencoder for learning acoustic word embeddings:

$$\mathbf{c} = f(\mathbf{x}_{1:T}) \tag{56}$$

$$\hat{\mathbf{x}}_{1:T} = g(\mathbf{c}) \ , \tag{57}$$

where $f(\cdot)$ is a recurrent neural network encoder, such that $\mathbf{c}$ is taken to be the hidden state at the last time-step T. The decoder, $g(\cdot)$, is also a recurrent neural network where $\mathbf{c}$ is used to initialize the hidden state in order to reconstruct the input in an autoregressive fashion. The authors also propose a denoising autoencoder with masked inputs $f(\mathbf{x}_{1:T} \circ \mathbf{m}_{1:T})$. Similar RNN-based autoencoders have been explored in other work as well [110, 137].

Prior to this work, Kamper et al. [138, 228] proposed the *correspondence autoencoder* based on a non-recurrent architecture. This method relies on unsupervised term discovery with randomized algorithms [126] for extracting corresponding speech segment pairs for training. Thus, one segment is used as input and the other as reconstruction target. Because the paired segments do not necessarily have the same length, they need to be aligned with dynamic time warping. In more recent work, this need has been alleviated by adopting the sequence-to-sequence framework [121, 137]. Unlike the reconstruction tasks reviewed in the previous sections, models presented here all rely on an ℓ_2 loss, $\mathcal{L} = \|\hat{\mathbf{x}}_{1:T} - \mathbf{x}_{1:T}\|_2$.

Inspired by the work on semantic word embeddings for text [194], the sequence-to-sequence framework has also been adopted to implement speech-based versions of the skipgram and continuous bag-of-words models [48, 49]. In this setting, the input sequence $\mathbf{x}_{1:T}$ is divided into word segments, such that we have $\mathbf{x}_{t_0:t_1}, \ldots, \mathbf{x}_{t_{N-1}:t_N}$ and we refer to $\mathbf{x}_{t_{n-1}:t_n}$ as $\mathbf{x}_{(n)}$. Here, N is the number of segments in a given utterance, $t_0 = 1$ and $t_N = T$. Now, given a target word $\mathbf{x}_{(n)}$, the skipgram model is trained to predict neighboring words $\mathbf{x}_{(n+k)}$

where k is a non-zero integer, possibly negative. That is, instead of a single decoder, as in eq. 57, the skipgram model employs multiple decoders

$$\hat{\mathbf{x}}_{(n+k)} = g_k(\mathbf{c}) \ . \tag{58}$$

Conversely, the continuous bag-of-words model is trained to predict the target word from the neighboring words, so here multiple encoders sum over several offsets $\mathcal{K}$ to obtain $\mathbf{c}$:

$$\mathbf{c} = \sum_{k \in \mathcal{K}} f_k(\mathbf{x}_{(n+k)}) \tag{59}$$

The sequence-to-sequence models described above rely on speech segments corresponding to words which are obtained by forced alignment. Thus, since forced alignment requires labeled data, the models might be seen as weakly supervised. However, similar models have been explored for general audio and speech embeddings without the requirement for exact word boundaries [127, 257].

Contrastive learning has also been explored for global speech representation learning. Comparable to the skipgram model described above, Milde and Biemann [197] and Jati and Georgiou [128] define a binary contrastive task where a target segment is used to predict neighboring segments. For general audio representation learning, Jansen et al. [124] explore multiple pre-training tasks using a contrastive triplet loss. And prior to the widespread adoption of neural networks, Levin et al. [167] explore principal component analysis and Laplacian eigenmaps for learning fixed-sized acoustic embeddings.

6.3.6 *Variable-rate representations*

Some models learn local representations with a temporal resolution that is not proportional to the input resolution. Instead, these models learn representations at a variable rate (VAR, table 7). Thus, given a length T input, the output resolution is not known a priori.

A significant amount of work is motivated by learning a segmentation of the input signal. One approach is to apply a dynamic programming algorithm to segment quantized representations post hoc [45, 139]. Another option is to infer segment boundaries from the error signal of a predictive model [154, 192]. Finally, gated activations of recurrent neural network autoencoders have also been shown to correlate with phoneme boundaries [271]. In all these cases, the representation learning model still maintains a constant temporal resolution during training. As such, using the representations to obtain segment boundaries may be viewed more as a downstream task, than a model feature. However, the learned segment boundaries can be used to obtain variable-rate representations.

In practice, few models directly map the input to a variable-rate representation. Dieleman et al. [69] propose the *slow autoencoder* where

repeated quantized values of the learned representation are taken to belong to the same segment. Repetition is explicitly encouraged through a penalty term, but still, the variable-rate aspect is somewhat implicit.

An exception is the recently proposed *segmental contrastive predictive coding* (SCPC). With this approach, the model explicitly learns segment boundary indicators, which are used to downsample the representations during training [25, 26]. The same segmentation strategy has subsequently been applied to other models [64].

6.3.7 *Other work*

Most of the work presented so far fits neatly into the taxonomy presented in table 7. One exception is the problem-agnostic speech encoder (PASE) [217, 227] that combines multiple pre-training tasks. Furthermore, many of the presented models have been successfully applied to other use cases. For instance, wav2vec 2.0 and related models have been applied to learn cross-lingual and multi-lingual representations [61, 143, 231] and proven well-suited for concurrently learning with labeled data [258, 270].

6.4 PROBABILISTIC LATENT VARIABLE MODELS

Another prominent class of models are probabilistic latent variable models (LVMs). Before surveying their application to speech, we briefly review LVMs and their usual specification when applied for representation learning in general. We disregard any specific temporal notation without loss of generality as, for instance, $\mathbf{x}$ can refer to a sequence as well as an image. Thereafter, we introduce the framework of the variational autoencoder [147]. We focus on the different dependency structures in and between data and learned representations explored in the literature, in contrast to the more practical view on self-supervised models taken above.

6.4.1 *LVMs and inference*

Fundamental to LVMs is the assumption that the data is produced by a generative process that involves unobserved stochastic latent variables $\mathbf{z}$. An LVM aims to model this generative process to enable generation of new data $\mathbf{x}$ and inference of the latent variable associated with a given observed variable $\mathbf{x}$. For representation learning, the inference of latent variables is of primary interest. An LVM is defined by the observation model $p(\mathbf{x}|\mathbf{z})$, which defines the relationship between the observed and latent variables, and the prior $p(\mathbf{z})$, which defines the relationship among the latent variables [19]. An LVM models the data generative process via the joint observation and prior model

$p(\mathbf{x}, \mathbf{z})$ often referred to as the generative model. The likelihood of an LVM given an example $\mathbf{x}$ can be written as

$$\log p(\mathbf{x}) = \log \int p(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) \, d\mathbf{z} \ . \tag{60}$$

The latent variable associated with an observed variable can be inferred with e.g. Markov Chain Monte Carlo (MCMC) methods [198] or variational inference [132].

For representation learning, LVMs are commonly defined in the framework of the variational autoencoder (VAE) [147, 230] which will also be the main focus of our exposition. In the VAE framework, the observation model $p(\mathbf{x}|\mathbf{z})$ is parameterized using a deep neural network. This choice allows modeling complex and high-dimensional data but also makes the integral in eq. equation 60 analytically intractable. MCMC methods can be used to estimate it and the true

Table 8: A comprehensive overview of observation, prior and inference models for VAE type latent variable models with a single latent variable. The observation, prior and inference models may all belong to one or more of the categories listed under them as detailed in section 6.4. The types listed here serve as primitives from which more complex structures can be constructed including models with hierarchies of multiple latent variables. We indicate autoregressiveness (ARX and ARZ) using a "catch-all" notation $*$, e.g. $\mathbf{x}_{*:t-1}$. This serves to indicate that autoregressive dependencies can have different span including at the extremes the full sequence $\mathbf{x}_{1:t-1}$ and the last value $\mathbf{x}_{t-1}$.

TYPE		FORM
	OBSERVATION MODEL	
ARX	Autoregressive on $\mathbf{x}_t$	$p(\mathbf{x}_t|\mathbf{x}_{1:t-1})$
LOC	Local latent variable	$p(\mathbf{x}_t|\mathbf{z}_{1:t})$
GLB	Global latent variable	$p(\mathbf{x}_t|\mathbf{z})$
	PRIOR	
ARX	Autoregressive on $\mathbf{x}_t$	$p(\mathbf{z}_t|\mathbf{x}_{1:t-1})$
ARZ	Autoregressive on $\mathbf{z}_t$	$p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$
IND	Locally independent latent variable	$p(\mathbf{z}_t)$
GLB	Global latent variable	$p(\mathbf{z})$
	INFERENCE MODEL	
ARZ	Autoregressive on $\mathbf{z}_t$	$q(\mathbf{z}_t|\mathbf{z}_{1:t-1})$
FLT	Filtering	$q(\mathbf{z}_t|\mathbf{x}_{1:t})$
LSM	Local smoothing	$q(\mathbf{z}_t|\mathbf{x}_{t-r:t+r})$
GSM	Global smoothing	$q(\mathbf{z}_t|\mathbf{x}_{1:T})$
GLB	Global latent variable	$q(\mathbf{z}|\mathbf{x}_{1:T})$

model posterior $p(\mathbf{z}|\mathbf{x})$, but these methods are usually computationally expensive in this setting [198]. To counter this and make gradient-based maximum likelihood training feasible, the VAE instead employs variational inference [132]. It approximates the intractable true model posterior by introducing a variational posterior distribution $q(\mathbf{z}|\mathbf{x})$, also parameterized by a deep neural network. From eq. 60, via Jensen's inequality, this gives rise to a variational lower bound on the likelihood, also known as the evidence lower bound (ELBO).

$$\log p(\mathbf{x}) \geqslant \int q(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})} \, d\mathbf{z} \equiv \mathcal{L}_{\text{ELBO}} \ . \tag{61}$$

The bound can be efficiently evaluated and optimized with Monte Carlo (MC) estimation by sampling from $q(\mathbf{z}|\mathbf{x})$. Low-variance gradient estimates through the stochastic sampling are usually obtained via reparameterization of $q(\mathbf{z}|\mathbf{x})$ [147] although alternatives exist (e.g., inverse CDF sampling) [198]. The ELBO can also be rewritten as

$$\mathcal{L}_{\text{ELBO}} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z})\right] - D_{\text{KL}}\left(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right) \ , \tag{62}$$

where $D_{\text{KL}}\left(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)$ is the Kullback-Leibler (KL) divergence between variational posterior distribution and prior, and $\mathbb{E}\left[\log p(\mathbf{x}|\mathbf{z})\right]$ can be seen as a reconstruction loss.

In brief, LVMs of the VAE-type consist of a parameterized approximate posterior, $q(\mathbf{z}|\mathbf{x})$, an observation model, $p(\mathbf{x}|\mathbf{z})$, and a prior, $p(\mathbf{z})$. The joint observation model and prior form the generative model which can be efficiently sampled with ancestral sampling which entails first sampling a latent variable $\tilde{\mathbf{z}} \sim p(\mathbf{z})$ followed by sampling the observed variable conditioned on that latent variable, $\tilde{\mathbf{x}} \sim p(\mathbf{x}|\tilde{\mathbf{z}})$.

With reference to probabilistic coding theory, the approximate posterior is often referred to as the encoder and the observation model as the decoder [147, 230]. From a theoretical perspective, the encoder exists solely as the result of choosing to using variational inference to train the decoder rather than e.g. MCMC. As such, it is also referred to as the inference model. However, from a representation learning perspective, the encoder is essential as it can be used to efficiently obtain the representation $\mathbf{z}$ commonly used for downstream tasks. In practice, it is customary to use the mode of $q(\mathbf{z}|\mathbf{x})$ rather than a sample $\mathbf{z}$ [114]. It is still possible to evaluate and sample the true posterior distribution $p(\mathbf{z}|\mathbf{x})$ by applying MCMC methods such as Hamiltonian Monte Carlo on the decoder, but for computational reasons this is rarely done in practice.

We next review the applications of VAEs to speech. We first consider the choices of observation, prior and inference models. As for self-supervised models, we introduce attributes throughout the text. An overview of these is found in table 8. We provide a model taxonomy for selected LVMs in table 9. Supplementing the binary attributes, table 10 explicitly details the observation, prior and inference models of the same selection of LVMs considered in table 9. The

graphical models for a subset these are shown in figure 13. We refer to the original papers for additional details.

6.4.2 *Observation models*

A common choice for the observation model $p(\mathbf{x}|\mathbf{z})$ is to include an autoregressive dependency on the observed variable (ARX, table 8) that is, $p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \cdot)$ where $\cdot$ represents some dependency on the latent variable [45, 56, 85, 265]. This allows the latent representation to focus on correlations that cannot easily be predicted from the observed variable at previous time-steps [45]. In practice, the dependency on $\mathbf{x}_{1:t-1}$ is often assumed to be Markovian and hence only on $\mathbf{x}_{t-1}$. Another common choice is to depend on a local window $\mathbf{x}_{t-r:t-1}$ where $r > 1$ is an integer denoting some receptive field. We will take a dependency on $\mathbf{x}_{1:t-1}$ to mean any one of these choices unless otherwise specified.

While the autoregressive dependency might be important for learning a powerful generative model, it might not benefit the learned latent representations. Specifically encouraging the latent representation to discard correlations across the temporal dimension might degrade the quality of the latent representation. Furthermore, since such a decoder can perform quite well by simply solving an autoregressive prediction problem, similar to WaveNet [207], it can make the model

		Observation			Prior				Inference					
Model	Pub. Date	ARX	LOC	GLB	ARX	ARZ	IND	GLB	ARZ	FLT	LSM	GSM	GLB	HIE
VRNN [56]	2015 Jun.	✓	✓	✗	✓	✓	✗	✗	✓	✓	✗	✗	✗	✗
SRNN [85]	2016 May	✓	✓	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗
HMM-VAE [79]	2017 Mar.	✗	✓	✗	✗	✓	✗	✗	✓	✓	✗	✗	✗	✓
ConvVAE [113]	2017 Apr.	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗	✓	✓	✗
FHVAE [114]	2017 Sep.	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✓	✗	✓
VQ-VAE [265]	2017 Nov.	✓	✓	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗
BHMM-VAE [89]	2018 Mar.	✗	✓	✗	✗	✓	✗	✗	✓	✓	✗	✗	✗	✗
STCN [1]	2019 Feb.	✗	✓	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓
FDMM [142]	2019 Oct.	✗	✓	✓	✗	✓	✗	✓	✓	✓	✗	✗	✓	✓
ConvDMM [144]	2020 Jun.	✗	✓	✗	✗	✓	✗	✗	✓	✗	✓	✗	✗	✗

Table 9: Selected probabilistic latent variable models classified according the binary attributes defined specifically for them throughout section 6.4. The models are sorted according to the publication date of the first version on arXiv which might differ from the year indicated by the citation. This table supplements the classification provided in table 7 which uses binary attributes largely defined for self-supervised methods. See table 8 for a listing of the concrete probability distributions and conditionings that correspond to each of the attribute short-hands. The mathematical expressions of the observation, prior and inference models for the models considered here can be found in table 10. Figure 13 depicts corresponding graphical model for selected LVMs of this table.

prone to suffer from so-called posterior collapse. This problem arises when the approximate and true posterior distributions collapse into the prior which renders the representations non-informative [33, 250]. Notably, posterior collapse corresponds to a local minimum in the ELBO since the KL-divergence becomes zero, achieving it's minimal possible value.

Some works alleviate the problem with tricks like KL-annealing and free bits. KL-annealing introduces a hyperparameter to weigh the KL-divergence in the ELBO (eq. 62) and gradually increases it from some small number to 1 during the initial training phase. Free bits is another such trick that returns a zero gradient for KL-divergences smaller than some number to avoid pushing the posterior completely into the prior [33, 150, 250]. The VQ-VAE uses a quantized latent space that is not susceptible to posterior collapse *per se* [45, 265]. How to equip LVMs with powerful decoders while avoiding posterior collapse still remains an open problem.

Some LVMs do not use autoregressive observation models [79, 89, 113, 114, 142, 144]. These more closely follow the classical assumption of *local independence* which states that the observed variables are conditionally independent given the local (LOC, table 8) and/or global (GLB, table 8) latent variables [19]. However, this enforces the latent variable to encode many details about the observed variable to achieve a good reconstruction. This is in the opposite vein of contrastive self-supervised learning which allows the model to discard details in $\mathbf{x}_{1:T}$ that do not inform the training objective [13].

6.4.3 *Prior*

Priors $p(\mathbf{z})$ can be said to belong to one or more of four broad categories: Autoregressive dependency on the observed variable (ARX, table 8), autoregressive dependency on the latent variable (ARZ, table 8), locally independent (IND, table 8) and global (GLB, table 8). Priors that are autoregressive on the observed variable take the form $p(\mathbf{z}_t|\mathbf{x}_{1:t-1})$. This generally results in a slow-down of the generative process which may be of concern if a use-case of the model is data generation. It is also interesting to note that encourages the latent variables to model less dynamic behaviour in their stochastic transitions and instead rely more on the observed variable. Priors that are autoregressive on the latent variable take the form $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$ and enable stochastic temporal transitions similar to hidden Markov models but with potentially nonlinear transition functions [56, 85, 142, 144]. Locally independent priors $p(\mathbf{z}_t)$ are rarely applied to sequential latent variables since they make the prior latent dynamics independent of the value of previous latent variables. Models that do impose such priors on sequential latents are quite limited in their generative power, unless they learn the prior dynamics post-hoc as done in e.g. the VQ-VAE [45, 265].

Global latent variables are fundamentally limited in the amount of information they can encode. Hence, models usually use them, either in combination with another local latent variable, or only encode inputs of some limited, fixed length segment [113, 114, 142].

Many LVMs have priors that do not have an autoregressive dependency on the observed variable (table 8). This allows them to generate the latent sequence in full before generating the observed sequence. Furthermore, this permits conditioning past observed variables on future latent variables. Despite this, it is common to let $\mathbf{x}_t$ depend causally on $\mathbf{z}_{1:t}$ in the generative model. To our knowledge, so far no work has examined full (non-causal) dependence of $\mathbf{x}_{1:T}$ on $\mathbf{z}_{1:T}$ during generation where, for example, $\mathbf{z}_T$ can influence $\mathbf{x}_1$.

6.4.4 *Inference models*

Inference models of LVMs based on the VAE perform so-called amortized variational inference. Herein, a single inference model is shared between all observed examples $\mathbf{x}$ and used to infer the associated latent variables $\mathbf{z}$. For this reason, all inference models covered here are conditioned on the observed sequence in some way. Generally, the inference model can be seen as solving either a filtering or smoothing problem. In filtering (FLT, table 8), the latent variables are assumed to depend only on past and current values of the observed variable such that it takes the form $q(\mathbf{z}_t|\mathbf{x}_{1:t})$ [56, 144]. In global smoothing (GSM, table 8), this causal dependency is replaced with a dependency on all observed values and hence takes the form $q(\mathbf{z}_t|\mathbf{x}_{1:T})$ [85, 113]. Smoothing can also be done locally (LSM, table 8), where the latent variables then depend only on a local context $\mathbf{x}_{t-r:t+r}$ for some integer $r > 0$ [45, 265]. Compared to self-supervised models, where transformer encoders have gained popularity, it can be hypothesized that global smoothing offers a stronger case than local smoothing and filtering for representation learning. The inference model may also be used to infer a global latent variable (GLB, table 8 that theoretically encodes global information about $\mathbf{x}$ separate from any local latent variables. It must be included in the prior model but not necessarily directly conditioned in the observation model. Finally, the latent variable is often made to depend on its past inferred values in an autoregressive manner, that is, $q(\mathbf{z}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:t})$ for filtering and $q(\mathbf{z}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{1:T})$ for smoothing (ARZ, table 8) [56, 85]. Such autoregressive dependencies help the inference model better match the prior dynamics but notably are not employed in models that learn the prior dynamics post-hoc [45, 265].

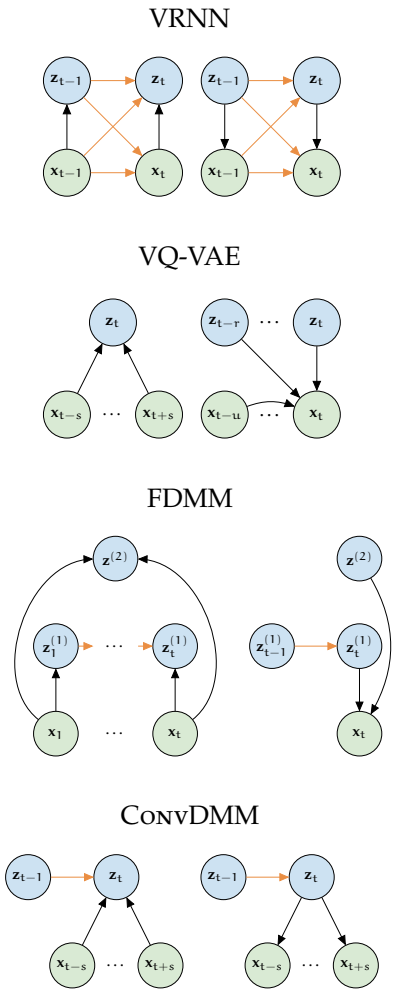


Figure 13: From the top, this figure shows the graphical models of the VRNN [56], VQ-VAE [265], FDMM [142] and ConvDMM [144] selected as they represent well the different attributes of table 8. The left-side graphs are the inference models $q(\mathbf{x}|\mathbf{z})$ and the right-side graphs are the generative models $p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$ which are the combined observation and prior models. Graphical models are a useful way to visualize probabilistic latent variable models that supplements the mathematical expressions for the conditional probability distributions. The expressions associated with the graphical models in this figure can be found in table 10. Arrows indicate transformations, which are usually implemented with neural networks. When colored orange, the associated parameters are shared between the inference and generative models. The VRNN is autoregressive on $\mathbf{x}_t$ and $\mathbf{z}_t$ in both inference and generative models. This allows sharing many parameters between the inference and generative models. VQ-VAE has an autoregressive observation model but uses a locally independent prior during training. An autoregressive prior can be learned after training. Inference is done with a locally smoothing convolutional encoder. The FDMM is illustrated in the so-called FDMM_iii configuration which uses a filtering inference model but the authors also examine global smoothing and a reverse filtering. The generative model is autoregressive on $\mathbf{z}_t^{(1)}$ and globally conditioned on $\mathbf{z}^{(2)}$. ConvDMM is locally smoothing and autoregressive on $\mathbf{z}_t$ in both generative and inference models.

Table 10: This table provides an overview of the observation, prior and inference models of a selection of the probabilistic latent variable models covered in this review. The models are presented as probability distributions and are per time-step assuming a factorization over time. The superscripts (1), (2) and (l) refer to a certain layer for models that include multiple latent variables in a hierarchy. The Viterbi algorithm is a dynamic programming approach for efficient posterior inference in latent variable models especially hidden Markov models [268].

MODEL	PUB. DATE	OBSERVATION	PRIOR	INFERENCE
VRNN [56]	2015 Jun.	$p(\mathbf{x}_t\|\mathbf{x}_{t-1},\mathbf{z}_t,\mathbf{z}_{t-1})$	$p(\mathbf{z}_t\|\mathbf{x}_{t-1},\mathbf{z}_{t-1})$	$q(\mathbf{z}_t\|\mathbf{x}_t,\mathbf{x}_{t-1},\mathbf{z}_{t-1})$
SRNN [85]	2016 May	$p(\mathbf{x}_t\|\mathbf{x}_{t-1},\mathbf{z}_t)$	$p(\mathbf{z}_t\|\mathbf{x}_{t-1},\mathbf{z}_{t-1})$	$q(\mathbf{z}_t\|\mathbf{x}_{1:T},\mathbf{z}_{t-1})$
HMM-VAE [79]	2017 Mar.	$p(\mathbf{x}_t\|\mathbf{z}_t^{(1)})$	$p(\mathbf{z}_t^{(1)}\|\mathbf{z}_t^{(2)})p(\mathbf{z}_t^{(2)}\|\mathbf{z}_{t-1}^{(2)})$	$q(\mathbf{z}_t^{(1)}\|\mathbf{x}_t)$ and Viterbi for $\mathbf{z}^{(2)}$
ConvVAE [113]	2017 Apr.	$p(\mathbf{x}_t\|\mathbf{z})$	$p(\mathbf{z})$	$q(\mathbf{z}\|\mathbf{x}_{1:T})$
FHVAE [114]	2017 Sep.	$p(\mathbf{x}_t\|\mathbf{z}_t^{(2)},\mathbf{z}_t^{(1)})$	$p(\boldsymbol{\mu}^{(2)})p(\mathbf{z}_t^{(2)}\|\boldsymbol{\mu}^{(2)})p(\mathbf{z}_t^{(1)})$	$q(\mathbf{z}_t^{(2)}\|\mathbf{x}_{1:T})\,q(\mathbf{z}_t^{(1)}\|\mathbf{x}_{1:T},\mathbf{z}_t^{(2)})$
VQ-VAE [265]	2017 Nov.	$p(\mathbf{x}_t\|\mathbf{x}_{t-u:t-1},\mathbf{z}_t)$	$p(\mathbf{z}_t)$ or $p(\mathbf{z}_t\|\mathbf{z}_{t-r:t-1})$	$q(\mathbf{z}_t\|\mathbf{x}_{t-s:t+s})$
STCN [1]	2019 Feb.	$p(\mathbf{x}_t\|\mathbf{z}_t^{(1:L)})$	$\prod_{l=1}^{L-1} p(\mathbf{z}_t^{(l)}\|\mathbf{x}_{t-r_l:t-1},\mathbf{z}_t^{(l+1)})$	$\prod_{l=1}^{L-1} q(\mathbf{z}_t^{(l)}\|\mathbf{x}_{t-r_l:t},\mathbf{z}_t^{(l+1)})$
FDMM [142]	2019 Oct.	$p(\mathbf{x}_t\|\mathbf{z}^{(2)},\mathbf{z}_t^{(1)})$	$p(\mathbf{z}^{(2)})\prod_t p(\mathbf{z}_t^{(1)}\|\mathbf{z}_{t-1}^{(1)})$	$q(\mathbf{z}^{(2)}\|\mathbf{x}_{1:T})\,q(\mathbf{z}_t^{(1)}\|\mathbf{x}_t,\mathbf{z}_{t-1}^{(1)})$
ConvDMM [144]	2020 Jun.	$p(\mathbf{x}_{t-r:t+r}\|\mathbf{z}_t)$	$p(\mathbf{z}_t\|\mathbf{z}_{t-1})$	$q(\mathbf{z}_t\|\mathbf{x}_{t-r:t+r},\mathbf{z}_{t-1})$

6.4.5 *Multiscale and hierarchical models*

Some work has explored using a hierarchy of latent variables (HIE, table 9). For instance, this allows encoding the inductive bias that speech contains information at different temporal scales by letting the latent variables operate at different temporal scales [114]. Khurana et al. [142] propose using a temporal latent variable along with a global latent variable. Recent work has focused on learning a deeper latent hierarchy with up to five latent variables [1]. The latents have no autoregressive temporal dependencies but instead depend on latent variables further up in the hierarchy with all latent variables operating at the same temporal scale. Interestingly, this effort within LVMs towards separating learned features into representations at different time scales or learning hierarchical representations stands in contrast to work on self-supervised learning where focus is either on learning local or global representations.

6.4.6 *Other work*

Before the introduction of the VAE, models such as deep belief networks (DBN) [107] built from stacks of restricted Boltzmann machines [83, 249] were popular. For instance, Lee et al. [166] show the feasibility of using a two-layered DBN for discovering acoustic units of

speech while Deng et al. [66] show that a DBN can learn a binary coding of spectrograms that have higher signal-to-noise ratio than classical vector-quantization techniques for speech coding. DBNs are however notoriously tricky to optimize requiring the use of expensive MCMC sampling techniques for inference or resort to biased gradient estimates [82, 106]. Non-neural approaches include Bayesian non-parametric models which can be used for unsupervised acoustic unit discovery often via a Dirichlet process prior which has the appealing property of automatically inferring the number of acoustic units [102, 165, 205]. Finally, Gaussian mixture models have been used to form so-called universal background models from which acoustic representations can be extracted [125].

6.5 EVALUATION PROCEDURES

So far, we have discussed qualitative differences between representation learning models. Next, we describe and discuss common evaluation procedures. First, we highlight efforts to standardize evaluation within the field, and then we focus on the specific evaluation procedures used in prior work.

6.5.1 *Challenges and benchmarks*

The zero resource speech challenge (ZeroSpeech) has played an important role in unsupervised representation learning over the years [75–78, 267]. The first edition took place in 2015 and has been followed up with challenges in 2017, 2019, 2020 and 2021. Each challenge has a different theme and provides corresponding training data and evaluation procedures. The evaluation procedures are typically intrinsic in nature. That is, there is no need for training additional classifier or regression models to compare the representation learning models. While this type of evaluation is typically fast to compute and yields consistent results, it is unclear how these metrics correlate with performance on downstream tasks.

Recently, new collections of downstream benchmark tasks have been introduced. The SUPERB benchmark [280] gathers multiple tasks grouped into categories such as *recognition*, *detection*, *semantics*, *speaker*, *paralinguistics* and *generation*. SUPERB reuse many existing tasks, but offers a standardized way to train the downstream models. Unlike the recent ZeroSpeech challenge, there are no restrictions on the data and the computational budget used to train the models. This may favor models trained with massive computational resources, rather than those presenting new innovations. With a more narrow scope than SUPERB, another recently proposed set of benchmark tasks is SLUE, which focuses on spoken language understanding (SLU) [246].

6.5.2 *Extrinsic evaluation*

As highlighted in the introduction, the goal of representation learning is to improve downstream tasks. Thus, most speech representation models are evaluated on their ability to solve such extrinsic tasks. When labeled data for a given task is limited and difficult to obtain, this is of particular interest. We discuss downstream tasks for evaluation in the following.

6.5.2.1 *Automatic speech recognition*

For models that learn local representations, automatic speech recognition is the *de facto* standard benchmark task (ASR, table 11). It is not surprising that contextualized representations are beneficial to this task given the importance of contextual information. The word error rate (WER) of an end-to-end speech recognition model has been shown to increase from 27% to 60% when context is removed [30]. This makes speech recognition an attractive evaluation task as it benefits from both acoustic and semantic features. Furthermore, speech recognition is an important prerequisite for many other downstream tasks that rely on text input.

However, many different datasets are used for downstream speech recognition (e.g., [87, 213, 220]), which makes comparison difficult. This is further complicated by variation in complexity and optimization of the downstream model. When a pre-trained representation learning model is fine-tuned, one has to decide on a fine-tuning strategy. When the model is not fine-tuned, a simple linear classifier is rarely sufficient for speech recognition which warrants a more complex architecture that can exploit temporal patterns. Finally, the choice of modeling unit (i.e., phonemes, characters, subwords or words) and the corresponding evaluation metric (i.e., PER, CER or WER) can further complicate comparison.

6.5.2.2 *Other downstream tasks*

For some models, speech recognition is not a meaningful evaluation task. Speech recognition requires a temporal input which excludes models that learn global representations. Such models may learn features related to speaker identity to which a speech recognition model should ideally be invariant. Instead, it is possible to probe a global representation for specific information, often relating to speaker, by choosing a more narrow downstream task. These include speaker verification [114, 142, 197], speaker identification [52, 128, 174, 206], dialect classification [142], emotion recognition [217, 280] and gender classification [166] (SPK, table 11). For local representations, phoneme classification is very common [47, 52, 113, 166, 175, 176]. Unlike phoneme-based speech recognition (i.e., phoneme

recognition), phoneme classification relies on manual or automated phoneme alignments. In table 11, phoneme classification is categorized as ASR.

While the tasks mentioned so far focus on features related to the speaker or acoustics of the signal, other tasks target semantic features. These tasks are typically referred to as SLU tasks (SLU, table 11) and include intent classification [31, 200, 280], slot filling [159, 280], sentiment analysis [55, 176], question answering [55], named entity recognition [31, 216, 246] and speech translation [17, 31, 50]. Cardiac arrest detection in emergency calls has also been used to evaluate speech representations [31].

6.5.3 *Intrinsic and non-parametric evaluation*

Intrinsic and non-parametric evaluation methods are usually less computationally demanding than extrinsic methods. Additionally, they typically yield more consistent results compared to extrinsic methods as they do not rely on a stochastic optimization algorithm. The methods we discuss in the following may require human annotations or occasionally rely on a pre-trained model. However, these efforts are one-off and do not have to be repeated when evaluating new representation learning models.

6.5.3.1 *Self-supervised pre-training metrics*

A straight-forward way to evaluate a representation learning model is to use the pre-training objective or a metric derived from the pre-training task. However, it is very rarely the case that performance on the pre-training task correlates positively with the usefulness of the representations for downstream tasks. In contrast, the opposite might be true. For masking-based models, we will usually see that reconstruction error will go down, if we use a less aggressive masking policy. However, this will probably also lead to lower quality representations as the task becomes increasingly easy. On the other hand, if too much of the input is masked, the model might be unable to learn anything and collapse to a trivial solution. Such metrics are better used as preliminary guidance on how to configure the pre-training task. For example, a model's ability to identify the correct positive sample from negative samples k steps into the future can be used to find a reasonable hyperparameter setting without using downstream tasks (e.g., as in [206]).

6.5.3.2 *LVM likelihood*

Analogous to the pre-training metric of self-supervised models, the lower bound on the likelihood (ELBO) defined in eq. 61 is a central intrinsic metric used for LVMs (LKH, table 11). Similar to pre-training

Table 11: An overview of evaluation methods for selected models, challenges and benchmarks. The models are the same as in table 7. **asr** means that automatic speech recognition or phoneme classification is used for evaluating the learned representations, **spk** that a speaker-based task is used, **slu** that a spoken language understanding task is used, **lkh** means that the model's likelihood is used, **abx** that an abx task is used and **oth** means that another intrinsic measure or task was used (data generation, query-by-example, etc.).

	Extrinsic			Intrinsic		
Model	ASR	SPK	SLU	LKH	ABX	OTH
Self-supervised models						
Audio W2V [54]	✗	✗	✗	✗	✗	✓
Speech2Vec [49]	✗	✗	✗	✗	✗	✓
Unspeech [197]	✓	✓	✗	✗	✗	✗
CPC [206]	✓	✓	✗	✗	✗	✗
PASE [217]	✓	✓	✗	✗	✗	✗
APC [52]	✓	✓	✗	✗	✗	✗
wav2vec [241]	✓	✗	✗	✗	✗	✗
Mockingjay [176]	✓	✓	✓	✗	✗	✗
wav2vec 2.0 [13]	✓	✗	✗	✗	✗	✗
NPC [174]	✓	✓	✗	✗	✗	✗
DeCoAR 2.0 [172]	✓	✗	✗	✗	✗	✗
SCPC [25]	✓	✗	✗	✗	✗	✗
HuBERT [111]	✓	✗	✗	✗	✗	✗
Probabilistic latent variable models						
VRNN [56]	✗	✗	✗	✓	✗	✓
SRNN [85]	✗	✗	✗	✓	✗	✗
HMM-VAE [79]	✓	✗	✗	✗	✗	✓
ConvVAE [113]	✓	✓	✗	✗	✗	✓
FHVAE [114]	✓	✓	✗	✗	✗	✓
VQ-VAE [265]	✗	✗	✗	✗	✗	✓
BHMM-VAE [89]	✓	✗	✗	✗	✗	✓
STCN [1]	✗	✗	✗	✓	✗	✓
FDMM [142]	✓	✓	✗	✗	✗	✗
ConvDMM [144]	✓	✗	✗	✗	✗	✗
Challenges and benchmarks						
ZeroSpeech [75–78, 267]	✗	✗	✗	✗	✓	✓
SUPERB [280]	✓	✓	✓	✗	✗	✗
SLUE [246]	✓	✗	✓	✗	✗	✗

metrics discussed above, a high ELBO does not guarantee that the model learns useful representations. In fact, the ELBO can be maximized while the representations learned by $q(\mathbf{z}|\mathbf{x})$ remain completely uninformative [116].

The reconstruction loss and KL-divergence terms of the ELBO presented in eq. 62 are often monitored independently. A high reconstruction loss indicates that the combined inference and observation models have not learned to reconstruct the input well which indicates poor representation quality. On the other hand, a low reconstruction loss does not guarantee that the learned representation is good. This is especially true when employing powerful autoregressive decoders. The KL-divergence can be interpreted as a regularization loss which we seek to minimize. However, as discussed above, a KL-divergence of zero is minimal but indicative of posterior collapse which yields uninformative representations. Hence, it is necessary, but not sufficient, that an LVM exhibits nonzero KL-divergence for its representations to be useful.

Likelihoods can be difficult to interpret. This can make it hard to reason about the absolute size of the ELBO and its terms. For discrete data, the negative ELBO can be interpreted as the minimum average number of bits required to losslessly compress the discrete data with an entropy coding scheme optimized for $q(\mathbf{z}|\mathbf{x})$ [243, 259, 261]. This can make it easy to compare the ELBO with baseline audio codecs such as FLAC [59] or MP3 [34]. Likewise, the KL-divergence term can be interpreted as the average number of extra bits required to encode samples of $q(\mathbf{z}|\mathbf{x})$ using a code optimized for $p(\mathbf{z})$ rather than one optimized for $q(\mathbf{z}|\mathbf{x})$. Hence, the larger the KL-divergence, the more informative $\mathbf{z}$ must be of $\mathbf{x}$.

6.5.3.3 *Minimal-pair ABX tasks*

A common intrinsic evaluation procedure is *minimal-pair ABX tasks* (ABX, table 11) [239, 240]. Here, we will review the approach taken in the early editions of the ZeroSpeech challenge [75–78, 267].

Consider two sets $\mathcal{A}$ and $\mathcal{B}$. Each set contains representation segments corresponding to a specific phoneme triplet. These segments are extracted using a test set with access to phoneme annotations and alignments. In order for $\mathcal{A}$ and $\mathcal{B}$ to constitute minimal pairs, they can only differ in the central phoneme. For example, using characters instead of phonemes for illustration, $\mathcal{A}$ might contain segments corresponding to `cat` while $\mathcal{B}$ contains segments corresponding to `cut`. Thus, minimal pairs differ little in phonetic content, but can have very different semantic content. The ABX score, $S_{\text{ABX}}(\cdot)$, is defined as the probability that a segment $\mathbf{a} \in \mathcal{A}$ is closer to $\mathbf{x} \in \mathcal{A}/\mathbf{a}$ than $\mathbf{b} \in \mathcal{B}$

according to some distance function $D(\cdot)$. Specifically, ignoring sequence subscripts, we have

$$I(\mathbf{a}, \mathbf{b}, \mathbf{x}) = \mathbb{1}_{D(\mathbf{a},\mathbf{x}) < D(\mathbf{b},\mathbf{x})} + \frac{1}{2}\mathbb{1}_{D(\mathbf{a},\mathbf{x}) = D(\mathbf{b},\mathbf{x})} \tag{63}$$

$$S_{ABX}(\mathcal{A}, \mathcal{B}) = \frac{1}{N} \sum_{\mathbf{a}\in\mathcal{A}} \sum_{\mathbf{b}\in\mathcal{B}} \sum_{\mathbf{x}\in\mathcal{A}/\mathbf{a}} I(\mathbf{a}, \mathbf{b}, \mathbf{x}) \ , \tag{64}$$

where $N = |\mathcal{A}||\mathcal{B}|(|\mathcal{A}| - 1)$ and $\mathbb{1}_E$ is a binary indicator of the event E. Since the segments $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{x}$ are not necessarily of the same length, it is common to use dynamic time warping with cosine similarity or Kullback-Leibler divergence [202] as the distance function $D(\cdot)$. Notice that $S_{ABX}(\cdot)$ is non-symmetric but can be symmetrized as $\frac{1}{2}(S_{ABX}(\mathcal{A}, \mathcal{B}) + S_{ABX}(\mathcal{B}, \mathcal{A}))$. To compute it over the full test set, we first average over all contexts (e.g., `c_t`) for a given central phoneme pair (e.g., `a-u`), and then over all central phoneme pairs.

ABX tasks, like the one described above, have been widely used in prior work and in all editions of the ZeroSpeech challenge [75–78, 267]. However, none of the recent models listed in table 7 use it. One downside to the approach is the need for aligned and annotated data. When it is preferable to use data from the same domain as the training data, this can be particular difficult to obtain. However, the largest collection of unsupervised data for speech representation learning, LibriLight, provides automatically generated phonetic transcriptions for the purpose [136].

6.5.3.4 *Other intrinsic evaluation methods*

Although we do not cover them in the same detail as the model likelihood and the ABX task above, we want to highlight a few other options for intrinsic evaluation (OTH, table 11). We find several examples in the previous ZeroSpeech editions: In 2015 and 2017, a series of F-score metrics were used to measure the quality of spoken term discovery systems [180]. The 2019 edition, which was themed *text-to-speech without text*, also considered representation bitrate and assessed the quality of synthesized speech using human judges. And in the 2021 edition, new distance and scoring-based metrics are used to assess a models ability to pick up on lexical, syntactic and semantic language features.

Of course, a variety of different measures have also been used outside the ZeroSpeech challenge. A word discrimination task proposed by Carlin et al. [36] has been frequently used to evaluate spoken term discovery systems [121, 138, 140]. Here, the learned representations for word pair segments are compared using some distance function and a threshold value. In the same vein, query-by-example is used to evaluate how well learned representations for a given example match other examples in a database belonging to the same category [54, 242]. And similar to the standard approach for evaluating text-based word

embeddings, Chung et al. [49] use a series of word similarity tasks. Finally, Pasad et al. [215] use canonical correlation analysis to compare speech representations to text-based word embeddings, acoustic word embeddings and other layers of the same model.

LVMs are commonly evaluated in terms of their ability to generate novel data examples. In the original work, the VQ-VAE is exclusively evaluated based on this ability [265]. Data generation can be extended by informed manipulation of the latent variable $\mathbf{z}$ which probes the learned representations for how well they disentangle targeted abstract features of $\mathbf{x}$. An example of this is voice conversion [113, 114]. The FHVAE [114] learns local and global information separately. In a qualitative evaluation, the authors use this to change the gender of a given utterance to that of a reference utterance while keeping the semantic content intact.

6.6 DISCUSSION

Throughout the text we defined several binary attributes which are used to characterize a selection of models in table 7, 9 and 11. We discuss representation learning models based on this overarching model taxonomy below.

6.6.1 *From global to local*

In table 7, we see that the work on global representations within self-supervised learning precedes the work on local representations – note that the models are sorted according to arXiv publication date. However, we find that many of the core ideas underlying global and local representation models are the same. For global representations, we saw that masking was proposed for a denoising autoencoder [54], sequence-to-sequence models have been tasked with context prediction, [49] and contrastive training has been applied to learn unsupervised speaker embeddings [197]. All these methods are now popular for local representation learning.

While much of the work on global representations relies on fairly shallow network architectures [49, 54], recent work on local representations employs very deep transformer networks [13, 111]. Furthermore, the global representations are usually learned from small speech segments, typically comprising no more than a word, whereas full sentences are used for recent local representation learning. And whereas work on global representation learning has taken inspiration from Word2vec [194], the techniques used for learning local representations are inspired by contextualized word embeddings [68]. Thus, the gap between these two model classes is largely a product of the developments in related fields and the general increase in computational resources.

Furthermore, the distinction between global and local representations in this taxonomy is a result of how the authors initially presented their work. In fact, it might be feasible to extract local representations from a model that is originally formulated as a global representation model and vice versa. For the sequence-to-sequence models that use the last hidden state of the encoder as a global representation (i.e., $\mathbf{c} = \mathbf{h}_T$) one could instead use the entire hidden state sequence as as a local contextualized representation (i.e., $\mathbf{c}_{1:T} = \mathbf{h}_{1:T}$). Conversely, local representations might simply be averaged to obtain a global representation. This approach has in fact proven successful [280].

6.6.2 *Choosing a target*

As evident from the REC column in table 7, a model that relies on the VAE framework will by definition learn to reconstruct the input speech. This might not be ideal for representation learning as discussed earlier. In summary, to create a reconstruction that closely matches the original input, the model needs to encode e.g. noise, small phase-shifts and amplitude scaling which might all be considered of relatively low importance to a good representation. In contrast, the notion that one can derive an alternative target from the input data is central to self-supervised learning [211]. For LVMs, one might also choose to model another representation than the raw audio or simple surface features (e.g., a spectrogram). Although the work is limited, this direction has been explored. Khurana et al. [144] model self-supervised wav2vec representations with a convolutional Markov model. They show that the resulting representation is better than one learned directly on speech and the original wav2vec representation when used as input for phone classification and recognition.

6.6.3 *Representations beyond inference*

While predictive tasks are commonly used for self-supervised models (PRD, table 7), they are not directly compatible with the LVM training objective. However, an LVM prior with an autoregressive parameterization, $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$ or $p(\mathbf{z}_t|\mathbf{x}_{1:t-1})$ (ARX or ARZ, table 8), can be seen as predictive in the sense that it tries to match the approximate posterior, which conditions on $\mathbf{x}_t$, using only latent dynamics and past observed variables. Hence, the prior might be considered for feature extraction although it remains unclear whether a representation extracted from the prior is better suited for downstream tasks than one extracted from the posterior. Jones and Moore [131] examine the importance of the prior in the VQ-VAE and show that the ability of this model to estimate densities $p(\mathbf{x}_{1:T})$ lies solely in its prior. Hidden units from the observation model have also been used as downstream

task features in previous work [144]. Similarly, Chorowski et al. [47] explore representations beyond the latent variable.

6.6.4 *Masking and missing data*

Masking (MSK, table 7) may also help to improve the representations learned with VAEs. Masking in VAEs has already been explored in the literature, but only in the context of missing data imputation. Here, $\mathbf{x}$ is only partially observed, and is often represented as a segmentation into observed and missing parts and a mask $\mathbf{m}$ indicating where the data is missing. The model is then trained to infer the latent variable from the observed part of the data and to reconstruct the missing part of the data, marginalizing out the missing data. Previous work has largely focused on the ability of these models to yield high-quality imputations within the tabular and image data domains, without probing for the effects on the learned latent representation [119, 189]. The idea of using VAEs to impute missing data was already examined in the seminal paper by Rezende et al. [230], but here the model was trained with completely observed data and merely used to impute data in an iterative sampling approach post hoc leaving the learned representations unchanged.

6.6.5 *Representation learning moving forward*

When it comes to representation learning, self-supervised models have been the primary force in pushing the field forward. As seen in table 11, speech recognition is the most common evaluation task, and it is in this area that the biggest advances have been made. We already addressed the success of models like wav2vec 2.0 in the context of end-to-end speech recognition [37, 92, 93] which reaches competitive performance on the widely used LibriSpeech dataset [13] when fine-tuned on 10 minutes of labeled data. Even without labeled data, wav2vec 2.0 can be used to learn a good speech recognition model [10].

On the other hand, the quality of the learned representations is not always probed for LVMs. Instead, work on these models is sometimes more focused on the generative capabilities. The VQ-VAE is perhaps the most successful case in this regard; the introduction of a quantized latent space, that allows for learning a simple autoregressive prior, proved instrumental for generating coherent speech. However, even here, models like wav2vec 2.0 have proven superior as a source for learning quantized units [163].

Recent successful approaches build upon the model and method of wav2vec 2.0. That is, deep transformer models combined with masking [41, 111, 270]. This development mirrors years of rapid progress in masked language modeling within natural language processing [58,

68]. We expect to see this development continue for unsupervised neural speech representation learning.

6.7 CONCLUSION

We reviewed unsupervised representation learning for speech, focusing on two primary categories: self-supervised methods and probabilistic latent variable models. Inspired by the development of self-supervised learning and the dependency structures of latent variable models, we derived a comprehensive model taxonomy. We then reviewed the evaluation of unsupervised speech representations. Finally, based on the model taxonomy, we discussed differences between models from the two categories and potential interesting avenues of future research.

ACKNOWLEDGMENTS

This work was partially supported by the Innovation Fund Denmark via the Industrial PhD Program (grant no. 0153-00167B and no. 8053-00184B). Christian Igel acknowledges support by the Pioneer Centre for AI (The Danish National Research Foundation, grant no. P1).

7

ON SCALING CONTRASTIVE REPRESENTATIONS FOR LOW-RESOURCE SPEECH RECOGNITION

ABSTRACT

Recent advances in self-supervised learning through contrastive training have shown that it is possible to learn a competitive speech recognition system with as little as 10 minutes of labeled data. However, these systems are computationally expensive since they require pre-training followed by fine-tuning in a large parameter space. We explore the performance of such systems without fine-tuning by training a state-of-the-art speech recognizer on the fixed representations from the computationally demanding wav2vec 2.0 framework. We find performance to decrease without fine-tuning and, in the extreme low-resource setting, wav2vec 2.0 is inferior to its predecessor. In addition, we find that wav2vec 2.0 representations live in a low dimensional subspace and that decorrelating the features of the representations can stabilize training of the automatic speech recognizer. Finally, we propose a bidirectional extension to the original wav2vec framework that consistently improves performance.

7.1 INTRODUCTION

Unsupervised learning for automatic speech recognition (ASR) has recently gained significant attention [12, 13, 50, 52, 53, 144, 217, 241, 273]. While the majority of work has focused on learning representations encoding the input for downstream tasks [12, 50, 52, 53, 144, 217, 241], the most promising results have been achieved with the wav2vec 2.0 framework (Fig. 14) where a pre-trained model is fine-tuned for speech recognition. However, these models are computationally expensive due to the large amount of memory intensive transformer layers. This contradicts the promise of easily applying these representations for new ASR models on low resource languages [13].

In contrast to wav2vec 2.0, its predecessor (Fig. 15) does not require fine-tuning as learned representations are used directly as input for an ASR model [241]. In addition, the pre-trained model has an order of magnitude fewer parameters than the large configuration of wav2vec 2.0. Because the frameworks are very similar, it seems obvious that representations extracted from wav2vec 2.0 would also be suitable input for training an ASR model. Training on extracted representations offers a light-weight alternative to the computationally expensive fine-tuning procedure described in [13].

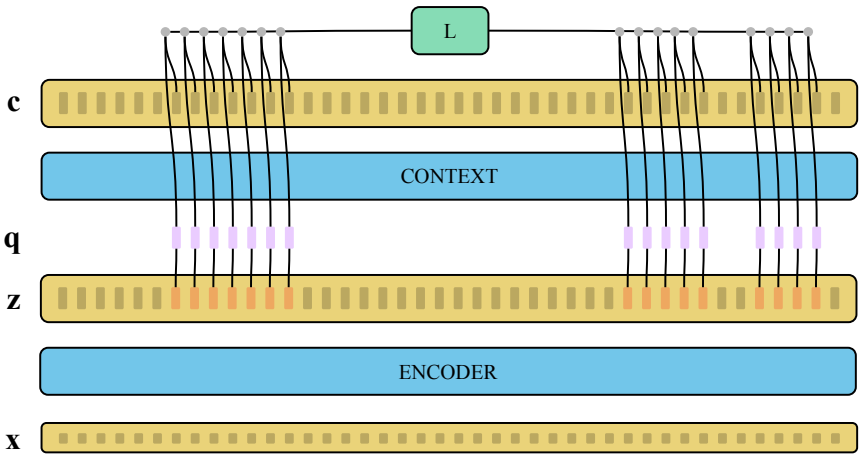


Figure 14: The wav2vec 2.0 framework [13]. The model is trained to identify the correct quantized target corresponding to the masked latent representations. The two proposed configurations have 95 and 317 million parameters respectively.

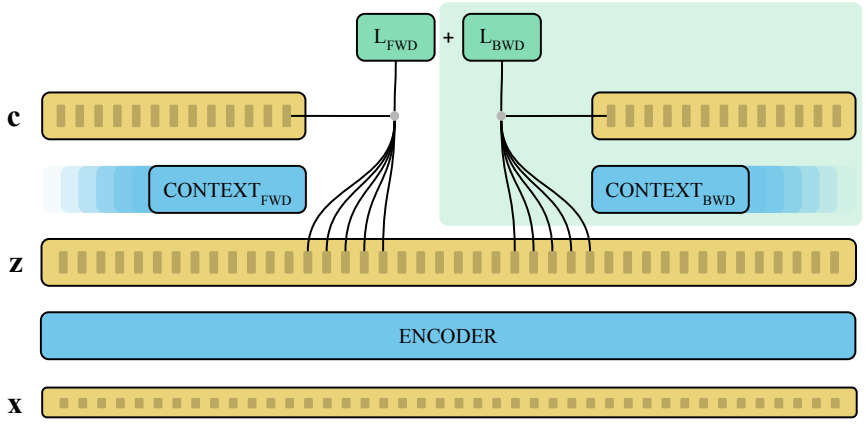


Figure 15: The wav2vec framework [241] extended with a backward context network (shaded area). The two context networks are independent, but are trained jointly with a shared encoder. The original model has 33 million parameters, while our extended model only has 18 million.

We study how representations from the two versions of the open-source wav2vec framework compare when used as input for low-resource end-to-end speech recognition. We also propose a bidirectional extension to the original wav2vec framework that, similar to wav2vec 2.0, can use the entire latent sequence to learn contextualized representations. Our contributions are as follows:

1. We find that ASR models trained on the wav2vec 2.0 representations often end up in poor local minima. Decorrelating the feature space dimensions with PCA alleviates the training issues.
2. We provide an overview of ASR models trained on contrastive representations from publicly available models. Despite using a strong ASR model, performance is heavily degraded compared to fine-tuning for wav2vec 2.0. When given only 10 minutes of training data, the original wav2vec model outperforms wav2vec 2.0.
3. We propose a bidirectional extension to the original wav2vec framework. Bidirectionality consistently improves performance of ASR models trained on the representations compared to representations from unidirectional baseline models.

7.2 CONTRASTIVE LEARNING FOR SPEECH

7.2.1 *wav2vec*

In the wav2vec framework, a PCM signal $\mathbf{x} \in \mathbb{R}^T$ of sequence length T is mapped to a sequence of latent representations $\mathbf{z} = \textsc{encode}(\mathbf{x}) \in \mathbb{R}^{U \times D}$ where U is the downsampled sequence length ($U = T/160$) and D is the dimensionality of the latent representation. This latent representation depends only locally on $\mathbf{x}$ with $\textsc{encode}(\cdot)$ parameterized by a convolutional neural network. The latent sequence $\mathbf{z}$ is fed to a context network to produce the representations used as input features for the downstream task $\mathbf{c} = \textsc{context}(\mathbf{z}) \in \mathbb{R}^{U \times D}$. In wav2vec, the context network is also convolutional but recurrent neural networks are an obvious alternative.

The model is trained with a contrastive loss function inspired by *contrastive predictive coding* [206] that maximizes the similarity between a contextualized representation $\mathbf{c}_u$ and the k'th future latent representation $\mathbf{z}_{u+k}$ through a learned step-specific affine transformation $\mathbf{H}_k \in \mathbb{R}^{D \times D}$:

$$\textsc{sim}_k(\mathbf{z}_i, \mathbf{c}_j) = \log(\sigma(\mathbf{z}_i^T \mathbf{H}_k \mathbf{c}_j)) \tag{65}$$

$$L_k(\mathbf{z}, \mathbf{c}) = -\sum_{i=1}^{U-k} (\textsc{sim}_k(\mathbf{z}_{i+k}, \mathbf{c}_i) \sum_{d \in \mathcal{D}} \textsc{sim}_k(-\mathbf{z}_d, \mathbf{c}_i)) \tag{66}$$

where $\sigma(\cdot)$ denotes the standard logistic function and $\mathcal{D}$ is a set of randomly sampled integers $d \sim \mathcal{U}\{1, U\}$ for indexing distractor samples $\mathbf{z}_d$. The total loss is defined as the sum over all K temporal offsets, $L = \sum_{k=1}^{K} L_k$. For further details on the wav2vec architecture and training procedure see [241].

7.2.2 *wav2vec 2.0*

Similar to the first version, wav2vec 2.0 also employs an encoder and a context network, but uses a coarser downsampling ($U = T/320$) in the encoder and a transformer-based context network [266]. In addition, a quantization network is used to learn a latent target sequence $\mathbf{q} = \text{QUANTIZE}(\mathbf{z}) \in \mathbb{R}^{U \times D}$. Before feeding $\mathbf{z}$ to the context network, approximately half of the U time steps are *masked* (i.e., replaced) by a learned D-dimensional vector. Given a context representation $\mathbf{c}_u$ corresponding to a masked latent vector $\mathbf{z}_u$, the model is trained to distinguish the quantized target $\mathbf{q}_u$ from distractors $\mathbf{q}_d$ sampled uniformly from the other masked time steps. The set of distractor indices $\mathcal{D}$ includes the target index u:

$$\text{SIM}(\mathbf{q}_i, \mathbf{c}_j) = \frac{\mathbf{q}_i^T \mathbf{c}_j}{\|\mathbf{q}_i\| \|\mathbf{c}_j\|} \tag{67}$$

$$L_u(\mathbf{q}, \mathbf{c}_u) = -\log \frac{e^{\text{SIM}(\mathbf{q}_u, \mathbf{c}_u)/\kappa}}{\sum_{d \in \mathcal{D}} e^{\text{SIM}(\mathbf{q}_d, \mathbf{c}_u)/\kappa}} \tag{68}$$

where κ is a constant temperature. The total loss is obtained by summing over all masked time steps. The model is also trained with an entropy-based diversity loss that encourages equal use of the quantized representations. The masking procedure allows for a context network consisting of multiple transformer layers that incorporate information from the full sequence instead of only time steps prior to u. Thus, the masking feature is key in order to be able to use an architecture well suited for fine-tuning.

7.3 BIDIRECTIONAL EXTENSION

The context network of the original wav2vec only uses information prior to the offset latent vector $\mathbf{z}_{i+k}$. This avoids collapsing to a trivial solution and allows for online processing of streaming data. In contrast, wav2vec 2.0 requires the complete sequence at once as input to the transformer-based context network. If we consider this setting where online processing is not required, the original wav2vec model can be extended with an additional context network that operates backward from time step U to 1. To train the backward network, the loss in equation 66 is adapted by replacing $\mathbf{z}_{i+k}$ with $\mathbf{z}_{i-k}$. The total loss is obtained by summing the loss for the two context networks as illustrated in Fig. 15. The context networks are independent, but

trained jointly with the same encoder. The representations used for downstream tasks are the concatenation of the output from the two context networks.

7.3.1 *Data and pre-trained models*

The original wav2vec model was trained on the 960 hours from the LibriSpeech dataset [213]. We trained our bidirectional extension and baseline models on the same data. We used three pre-trained models from wav2vec 2.0: BASE, LARGE and VOX[1]. The LARGE model is a deeper and wider version of the BASE model. Both are trained on the 960 hour LibriSpeech. The VOX model is identical to the LARGE, but trained on 60.000 hours of speech from the LibriLight dataset [136] which is an in-domain extension of LibriSpeech providing large quantities of unlabeled data and a standardization of smaller subsets from the original 960 hours training data. We trained ASR models on the 10 minute, 1 hour and 10 hour subsets of LibriLight for all representation models.

7.3.2 *Training procedures*

In addition to bidirectionality, we propose to use few filters for the first layer in the encoder network and then incrementally increase the number of filters as the temporal resolution is lowered by striding. This significantly lowers the memory footprint of the encoder by avoiding large representations while the temporal resolution is high. Thus, our encoder uses six 1D-convolutions with number of filters set to (64, 128, 192, 256, 512, 512), kernel sizes (10, 8, 4, 4, 4, 1), and strides (5, 4, 2, 2, 2, 1). With a constant filter size of 512, memory consumption would be 4.6 times higher. Each convolutional layer is followed by a group normalization layer with 32 groups [276] and ReLU nonlinearity clipped at the value of 5. Instead of using convolutions as in wav2vec, we used four LSTM layers [109] with 512 units each for the context network. We sampled 120 seconds of audio for each batch and trained the model for 8 epochs on LibriSpeech. We used Adam [146] with a fixed learning rate of $3 \cdot 10^{-4}$ for the first half of training after which it was decayed to $5 \cdot 10^{-5}$. We use $K = 12$ offsets and sample 10 distractors. (i.e., $|\mathcal{D}| = 10$).[3]

For the ASR model, we used the architecture from [30] trained with a connectionist temporal classification loss [93], which has shown state-of-the-art results on the small Wall Street Journal dataset [220]. The original model uses three layers of 2D-convolutions followed by 10 bidirectional LSTM layers with skip-connections and 320 units

1 https://github.com/pytorch/fairseq
2 The model is trained on 16 GPUs, but training time is not stated.
3 https://github.com/corticph/scaling-contrastive

	10 min.		1 hour		10 hour					
Name	**clean**	**other**	**clean**	**other**	**clean**	**other**	**PCA**	**size** D	**params**	**GPU days**
Log mel-spectrogram	99.6	99.7	66.5	82.0	33.8	57.5	No	80	-	-
wav2vec [241]	71.7	82.5	43.1	61.9	24.0	45.8	No	512	33M	?[2]
wav2vec 2.0 [13]										
BASE	79.5	87.5	38.7	**53.6**	**14.9**	**28.4**	Yes	768	95M	102.4
LARGE	91.0	96.1	50.2	66.5	20.5	37.5	Yes	1024	317M	294.4
VOX	94.2	97.2	42.4	56.8	16.4	29.8	Yes	1024	317M	665.6
Our work:										
LSTM-UD-512	69.9	81.9	41.5	60.9	23.8	45.2	No	512	9.6M	3.8
LSTM-UD-2x512	69.2	81.2	41.0	61.0	23.2	44.9	No	1024	18M	9.5
LSTM-BD-2x512	**65.2**	**77.4**	**37.9**	56.5	21.0	42.1	No	1024	18M	9.4

Table 12: Word error rates on the clean and other test sets of LibriSpeech for ASR models trained with representations extracted from wav2vec, wav2vec 2.0 and the authors' proposed models. Results are without an external language model. *GPU days* denotes training time multiplied by the number of GPUs.

each. We replaced the 2D-convolutions with 1D-convolutions as there is no structure along the feature dimension of the learned representations. All 1D-convolutions used kernel size 3, had (640, 480, 320) units and strides (2, 1, 1). To account for the lower temporal resolution of the wav2vec 2.0 representation, strides were reduced to (1, 1, 1). We used the same optimizer and learning rate schedule as for the CPC models and batches were created by sampling up to 320 seconds of audio. The models were trained for 25k update steps on the 1 hour and 10 hour subsets, but only for 10k update steps on the 10 minute subset. Total training time was ~ 12 hours on a single GPU for 25k updates. Results reported for the 10 minute models are averages over the 6 separate subsets of LibriLight.

7.4 RESULTS

7.4.1 *Training with wav2vec 2.0 representations*

We found that ASR models trained on representations extracted from the wav2vec 2.0 models had a tendency to get stuck in poor local minima. After confirming that values of the learned features followed a reasonable distribution, and that tuning the learning rate did not solve the issue, we performed a principal component analysis (PCA) of the representations. We found that the wav2vec 2.0 representations generally exhibited a low *linear dimensionality*, that is, only few principal components are needed to explain the variance in the representations, see Fig. 16. Furthermore, the linear dimensionality of the representation decreased with model complexity and the amount of training data. Indeed, the two large models were also the ones that consistently failed, while the BASE model did converge on both the 1 hour and 10 hour subsets. Feature decorrelation has previously proven useful in speech classification tasks [288]. Training ASR models on the decorrelated feature space, without reducing the number of features, solved the initial training issues. To ensure that the mean normalization commonly performed prior to the PCA transformation was not responsible for resolving the issue, we performed an ablation experiment where we only used mean normalization on the raw features, but this did not alleviate training issues. Representations from our models and wav2vec did not benefit from decorrelation.

7.4.2 *Performance: wav2vec and wav2vec 2.0*

Surprisingly, the BASE representations consistently outperformed representations from the two larger models, indicating that the quality of the learned representations does not scale with model complexity for wav2vec 2.0. For the 1 hour and 10 hour subsets, representations from the VOX model led to better performance compared to the

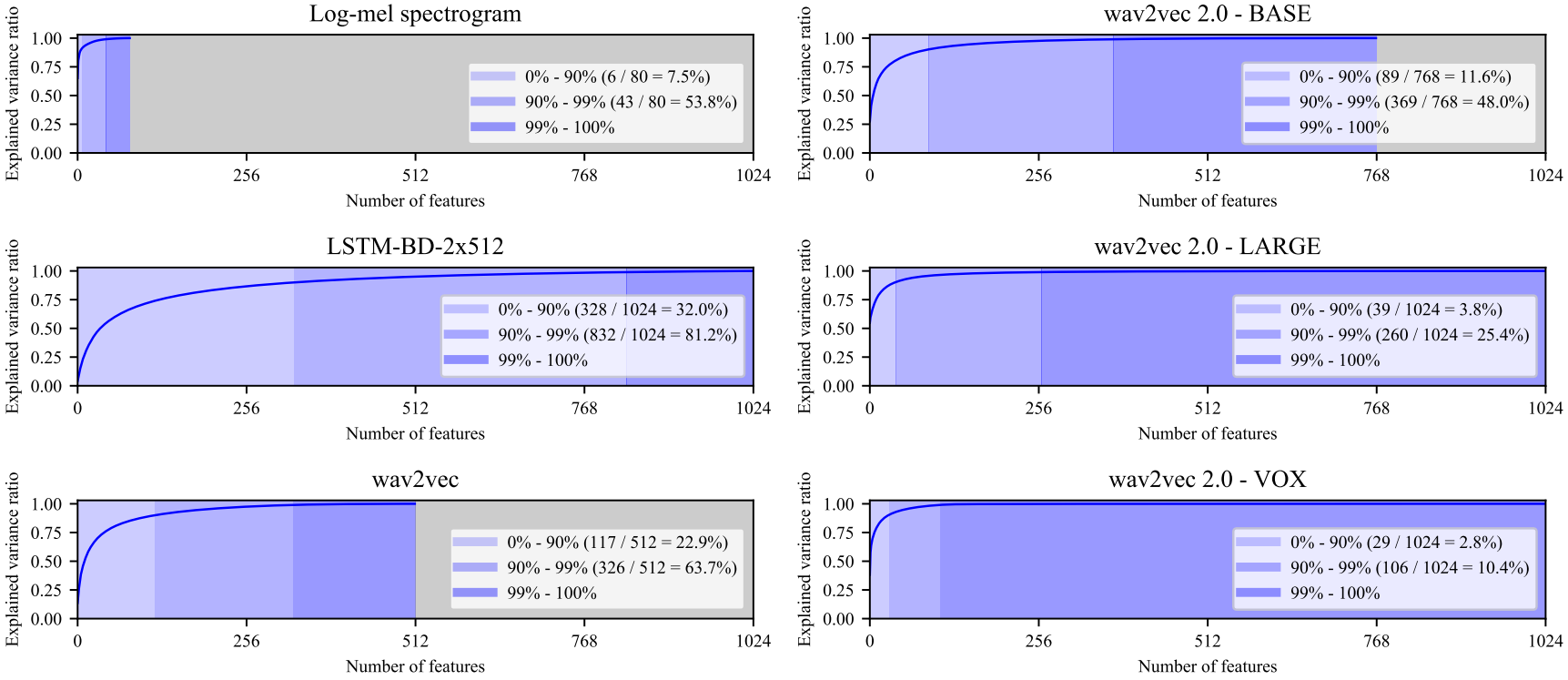


Figure 16: Explained variance ratio as a function of the number of features after decorrelating the feature space with PCA. The PCA transformation was computed on the 10 hour subset of LibriLight.

LARGE model showing the benefits of the large increase in training data. This tendency was blurred by the poor performance of both representations on the 10 minute subset. Compared to fine-tuning results without a language model in Appendix C of [13], performance is severely degraded despite a strong ASR model.

Focusing on the best performing wav2vec 2.0 representations from BASE, we observe a significant word error rate reduction for the 10 hour subset compared to wav2vec. As the amount of training data is reduced, so is the difference. For the 10 minute subset, the picture is reversed as the wav2vec representations performed better.

7.4.3 *Performance: Bidirectionality*

Our baseline model (LSTM-512), as expected, yielded representations on par with the wav2vec model. The bidirectional extension (LSTM-BD-2x512) consistently improved word error rate for all subsets. To ensure the improvement is not just a result of increased model complexity, we trained another model that also used two separate context networks, but both operating in the same direction (LSTM-UD-2x512). Although this model was slightly better than the baseline model, the bidirectional model was still superior across all subsets. Furthermore, the bidirectional model gave the best result on the clean test set when trained on the 1 hour subset and for both test sets when trained on the 10 minute subset.

7.5 CONCLUSIONS

We compared contrastive representations for ASR in the setting of limited training resources. We showed that representations from the wav2vec 2.0 framework live in a low dimensional subspace. Using PCA to decorrelate the features alleviated training issues for the speech recognizer. However, ASR models trained on the fixed wav2vec 2.0 representations still performed significantly worse than the fine-tuned versions from the original wav2vec 2.0 work. Representations from the first version of wav2vec, learned with a much lower computational cost, performed better than wav2vec 2.0 on the 10 minute subset, but were inferior on the 10 hour subset. We extended the original wav2vec framework with a context network operating backwards along the temporal dimension and confirmed that bidirectionality can improve speech representations used for ASR.

8

BENCHMARKING GENERATIVE LATENT VARIABLE MODELS FOR SPEECH

ABSTRACT

Stochastic latent variable models (LVMs) achieve state-of-the-art performance on natural image generation but are still inferior to deterministic models on speech. In this paper, we develop a speech benchmark of popular temporal LVMs and compare them against state-of-the-art deterministic models. We report the likelihood, which is a much used metric in the image domain, but rarely, and often incomparably, reported for speech models. To assess the quality of the learned representations, we also compare their usefulness for phoneme recognition. Finally, we adapt the Clockwork VAE, a state-of-the-art temporal LVM for video generation, to the speech domain. Despite being autoregressive only in latent space, we find that the Clockwork VAE can outperform previous LVMs and reduce the gap to deterministic models by using a hierarchy of latent variables.

8.1 INTRODUCTION

After the introduction of the variational autoencoder (VAE, [147, 230]) quickly came two temporal extensions for modeling speech data [56, 85]. Since then, temporal LVMs have undergone little development compared to their counterparts in the image domain, where LVMs recently showed superior performance to autoregressive models such as PixelCNN [208, 209, 237]. The improvements in the image domain have been driven mainly by top-down inference models and deeper latent hierarchies [43, 185, 250, 264]. In speech modeling however, autoregressive models such as the WaveNet remain state-of-the-art [207].

To compare and develop LVMs for speech, we need good benchmarks similar to those in the image domain. Image benchmarks commonly compare likelihood scores, but research in the speech domain often omits reporting a likelihood [114, 207, 265] or report likelihoods that are incomparable due to subtle differences in the assumed data distribution [1, 56, 85, 114]. Without a proper comparison standard, it is difficult for the field of explicit likelihood models on speech to develop in a directed manner.

To advance the state of LVMs for speech, this paper (i) develops a benchmark for LVMs based on model likelihood, (ii) introduces a hierarchical LVM architecture without autoregressive decoder, (iii) com-

pares LVMs to deterministic counterparts including WaveNet, and (iv) qualitatively and quantitatively evaluates the latent variables learned by different LVMs based on their usefulness for phoneme recognition. We find that:

(I) State-of-the-art LVMs achieve likelihoods that are inferior to WaveNet at high temporal resolution but are superior at lower resolutions. Interestingly, we find that a standard LSTM [109] almost matches the likelihood of WaveNet.

(II) LVMs with powerful autoregressive decoders achieve better likelihoods than the non-autoregressive LVM.

(III) The expressiveness of LVMs for speech increases with a deeper hierarchy of stochastic latent variables, similar to conclusions within image modeling.

(IV) LVMs learn rich representations that are as good or better than Mel spectrograms for phoneme recognition also when using only 10 minutes of labeled data.

At a high level, this benchmark brings order to LVM model comparisons for speech and also provides useful reference implementations of the models[1]. Before presenting the results, we provide a brief survey of existing LVMs for speech in a coherent notation.

8.2 LATENT VARIABLE MODELS FOR SPEECH

LVMs formulated via the VAE framework continue to be of interest due to their ability to learn an approximation to the posterior distribution of assumed latent variables. The posterior is usually of a significantly reduced dimensionality compared to the input and lies close to a known prior distribution. Approximate posteriors are useful for tasks beyond generation such as semi-supervised learning [149] and anomaly detection [100].

In recent years, several complementary methods have been proposed to improve the expressiveness of VAEs. These include building more expressive priors via methods such as normalizing flows [229] and building a deeper hierarchy of stochastic latent variables such as the Ladder VAE [250]. In this research, we focus on the latter due to the recent breakthroughs in image modeling using VAEs without costly autoregressive dependencies on the observed variable [43, 185, 264].

Several works have applied LVMs to speech. Among the first contributions were the VRNN [56] and SRNN [85] which can be seen as conditional VAEs per timestep. Other recent LVMs include the FHVAE [114], which leverages an additional latent variable to capture global features, and Z-forcing [91], which resembles the SRNN but includes an auxiliary task in the latent space to increase its utilization.

1 github.com/anonymous/available-later

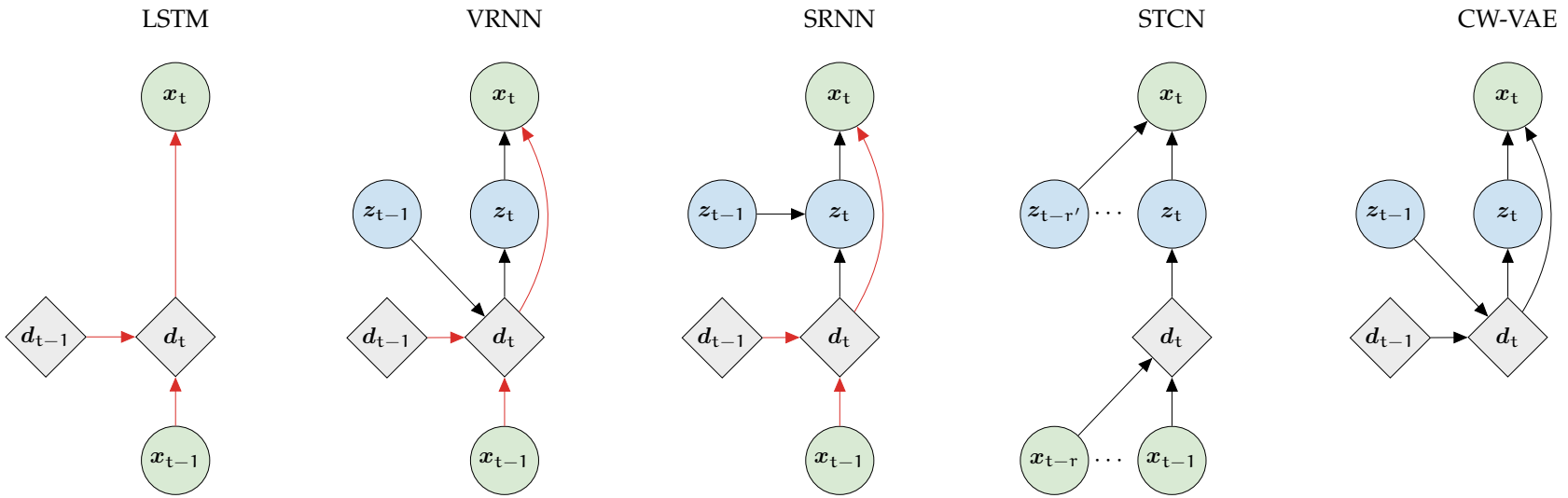


Figure 17: Generative models for a single time step of a deterministic autoregressive LSTM, the VRNN and SRNN as well as the STCN and CW-VAE both with a single layer of latent variables. Red arrows indicate purely deterministic paths from the output x_t to previous input $x_{<t}$ without passing a stochastic node. In an LSTM, information flows deterministically from previous observed variables to the next. The VRNN and SRNN use stochastic latent variables but also include deterministic skip connections from previous observed variables. The STCN is also autoregressive in x but does not use deterministic skip connections from x_{t-1} to x_t. The CW-VAE is not autoregressive on the observed variable which forces information to flow through the latent variables. We provide more elaborate graphical illustrations including inference models of the CW-VAE in figure 18 and of the VRNN, SRNN and STCN in appendix A.10.

The VQ-VAE [265] is a hybrid between an LVM and an autoregressive model which uses a quantized latent space to improve the quality of generated samples. The Stochastic WaveNet [162] and STCN [1] use WaveNet encoder and decoders and have latent variables that are temporally independent.

In this paper, we focus on the VRNN, SRNN and STCN. We exclude the Stochastic WaveNet as it is similar to STCN and achieves inferior likelihoods [1]. The FH-VAE, with disjoint latent variables and discriminative objective, Z-forcing, with an auxiliary task, and the VQ-VAE, with a quantized latent space and autoregressive prior fitted after training, all introduce significant changes to the original VAE framework and are also not included here.

Notably, all selected models have autoregressive generative models which let future observed variables be generated by conditioning on previously generated values. Inspired by recent progress in the image domain, we therefore formulate and benchmark a novel temporal LVM which does not rely on an autoregressive decoder. We do so by adapting to speech the hierarchical Clockwork Variational Autoencoder [238] recently proposed for video prediction.

8.2.1 *Sequential deep latent variable models*

The selected models are all sequential deep latent variable models trained with variational inference and the reparameterization trick [147]. They take as input a variable-length sequence $\boldsymbol{x}_{1:T} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T)$ with $\boldsymbol{x}_t \in \mathbb{R}^{D_x}$. We let $\boldsymbol{x}_{1:T}$ refer to the observed variable or a downsampled version of it. We will sometimes use $\boldsymbol{x}$ to refer to the sequence $\boldsymbol{x}_{1:T}$ when it is not ambiguous.

First, $\boldsymbol{x}_{1:T}$ is encoded to a temporal stochastic latent representation $\boldsymbol{z}_{1:T} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_T)$ with $\boldsymbol{z}_t \in \mathbb{R}^{D_z}$. This representation is then used to reconstruct the original input $\boldsymbol{x}_{1:T}$. The latent variable is assumed to follow some prior distribution $p(\boldsymbol{z}_t|\cdot)$ where the dot indicates that it may depend on latent and observed variables at previous time steps, $\boldsymbol{z}_{<t}$ and $\boldsymbol{x}_{<t}$ where $\boldsymbol{z}_{<t} := (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{t-1})$.

The models are trained to maximize a likelihood objective. The exact marginal likelihood is given by

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) \, d\boldsymbol{z} \ , \tag{69}$$

where $\boldsymbol{\theta}$ are parameters of the generative model. The exact likelihood is intractable to optimize due to the integration over the latent space. Instead, we introduce the variational approximation $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ to the true posterior. Via Jensen's inequality this yields the well-known evidence lower bound (ELBO) on the exact likelihood

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \geqslant \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right] =: \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \boldsymbol{x}) \tag{70}$$

which can be jointly optimized with respect to $\{\boldsymbol{\theta}, \boldsymbol{\phi}\}$ using stochastic gradient descent methods. We omit the $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ subscripts for the remainder of the paper. A graphical illustration of the models can be seen in figure 17. Additional graphical models can be found in appendix A.10.

8.2.2 *Variational recurrent neural network (VRNN)*

The VRNN [56] is essentially a VAE per timestep. At timestep t, the VAE is conditioned on the hidden state of a recurrent neural network (RNN), $\boldsymbol{d}_t \in \mathbb{R}^{D_d}$, with state transition $\boldsymbol{d}_t = f([\boldsymbol{x}_{t-1}, \boldsymbol{z}_{t-1}], \boldsymbol{d}_{t-1})$ where $[\cdot, \cdot]$ denotes concatenation. The VRNN uses a Gated Recurrent Unit (GRU, [44]) for f. The joint distribution factorizes over time and the latent variables are autoregressive in both the observed and latent space,

$$p(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{\leqslant t}) p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t}) \ . \tag{71}$$

The approximate posterior similarly factorizes over time,

$$q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} q(\boldsymbol{z}_t|\boldsymbol{x}_{\leqslant t}, \boldsymbol{z}_{<t}) \ . \tag{72}$$

From this, the ELBO for the VRNN is

$$\begin{aligned} \mathcal{L}(\boldsymbol{x}) = \mathbb{E}_{q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})} \Bigg[\sum_t \log p(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{\leqslant t}) \\ - \mathrm{KL}\left(q(\boldsymbol{z}_t|\boldsymbol{x}_{\leqslant t}, \boldsymbol{z}_{<t}) \,\|\, p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t})\right) \Bigg] \ . \end{aligned} \tag{73}$$

The VRNN uses diagonal covariance Gaussian distributions $\mathcal{N}$ for the prior and posterior distributions. We denote the output distribution of choice by $\mathcal{D}$.

$$q(\boldsymbol{z}_t|\boldsymbol{x}_{\leqslant t}, \boldsymbol{z}_{<t}) = \mathcal{N}\left(\boldsymbol{\alpha}_q(\boldsymbol{x}_t, \boldsymbol{d}_t)\right) \tag{74}$$

$$p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t}) = \mathcal{N}\left(\boldsymbol{\alpha}_p(\boldsymbol{d}_t)\right) \tag{75}$$

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{\leqslant t}) = \mathcal{D}\left(\boldsymbol{\beta}(\boldsymbol{z}_t, \boldsymbol{d}_t)\right) \ . \tag{76}$$

All sets of distributional parameters, $\boldsymbol{\alpha}_q, \boldsymbol{\alpha}_p$ and $\boldsymbol{\beta}$, are the outputs of densely connected neural networks which we notationally overload as functions in equations (74-76). It is common to refer to $\boldsymbol{\alpha}_q$ as the inference model or encoder and $\boldsymbol{\beta}$ as the decoder. Together with $\boldsymbol{\beta}$, the structural model $\boldsymbol{\alpha}_p$ forms the generative model.

Since the decoder is dependent on $\boldsymbol{d}_t$, the transition function f allows the VRNN to learn to ignore parts of or the entire latent variable and establish a purely deterministic transition from $\boldsymbol{x}_{t-1}$ to $\boldsymbol{d}_t$ (figure 17). This failure mode is commonly referred to as posterior collapse and is a well-known weakness of VAEs with powerful decoders [33, 250].

8.2.3 *Stochastic recurrent neural network (SRNN)*

The SRNN [85] is similar to the VRNN but differs by separating the stochastic latent variables from the deterministic representations (figure 17). That is, the GRU state transition is independent of $\boldsymbol{z}_{1:T}$ such that $\boldsymbol{d}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{d}_{t-1})$. With this, the joint $p(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T})$ can be written as for the VRNN in equation 71. The approximate posterior of $\boldsymbol{z}_t$ is conditioned on the full observed sequence,

$$q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = \prod_{t=1}^{T} q(\boldsymbol{z}_t|\boldsymbol{x}_{1:T}, \boldsymbol{z}_{t-1}) \ . \tag{77}$$

This is achieved by introducing a second GRU that runs backwards in time with transition $\boldsymbol{a}_t = g([\boldsymbol{x}_t, \boldsymbol{d}_t], \boldsymbol{a}_{t+1})$. While $p(\boldsymbol{x}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{\leqslant t})$ remains as in equation 76, we have

$$q(\boldsymbol{z}_t|\boldsymbol{x}_{1:T}, \boldsymbol{z}_{<t}) = \mathcal{N}(\boldsymbol{\alpha}_q(\boldsymbol{z}_{t-1}, \boldsymbol{a}_t)) \tag{78}$$

$$p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t}) = \mathcal{N}(\boldsymbol{\alpha}_p(\boldsymbol{z}_{t-1}, \boldsymbol{d}_t)) \ . \tag{79}$$

By inferring $\boldsymbol{z}_t$ conditioned on the full sequence $\boldsymbol{x}_{1:T}$, the SRNN performs smoothing. This has been noted to better approximate the true posterior of $\boldsymbol{z}_t$ which can be shown to depend on the full observed sequence [21]. Comparatively, the VRNN performs filtering.

8.2.4 *Stochastic temporal convolutional network (STCN)*

The STCN [1] is a hierarchical latent variable model with an autoregressive generative model based on WaveNet [207]. Contrary to VRNN and SRNN, the latent variables are independent in the sense that there are no transition functions connecting them over time. Instead, a latent $\boldsymbol{z}_t^{(l)}$ at layer l is conditioned on a window of observed variables $\boldsymbol{x}_{\mathcal{R}_t^{(l)}}$ via a WaveNet encoder. The windowed dependency introduced by the WaveNet is defined via the index set $\mathcal{R}_t^{(l)} = \{t - r_l + 1, \ldots, t\}$ where r_l is the receptive field of the encoder at layer l. The window size grows exponentially with the layer l. The joint can be written as

$$p(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\boldsymbol{z}_{\mathcal{R}_t^{(1)}}^{(1:L)}) \prod_{l=1}^{L} p(\boldsymbol{z}_t^{(l)}|\boldsymbol{x}_{\mathcal{R}_{t-1}^{(l)}}, \boldsymbol{z}_t^{(l+1)})$$

where $\boldsymbol{z}_t^{(L+1)} := \emptyset$ for notational convenience. The deterministic encoding is $\boldsymbol{d}_t^{(l)} = h(\boldsymbol{x}_{\mathcal{R}_t^{(l)}})$ where h is the encoder and $\boldsymbol{d}_t^{(l)}$ is extracted from the l'th layer similar to a Ladder VAE [250]. The approximate posterior is

$$q\left(\boldsymbol{z}_{1:T}^{(1:L)}|\boldsymbol{x}_{1:T}\right) = \prod_{t=1}^{T} \prod_{l=1}^{L} q\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{x}_{\mathcal{R}_t^{(l)}}, \boldsymbol{z}_t^{(l+1)}\right) \tag{80}$$

The factorized distributions are given as

$$q\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{x}_{\mathcal{R}_t^{(l)}}, \boldsymbol{z}_t^{(l+1)}\right) = \mathcal{N}\left(\boldsymbol{\alpha}_q^{(l)}\left(\boldsymbol{z}_t^{(l+1)}, \boldsymbol{d}_t^{(l)}\right)\right) \tag{81}$$

$$p\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{x}_{\mathcal{R}_{t-1}^{(l)}}, \boldsymbol{z}_t^{(l+1)}\right) = \mathcal{N}\left(\boldsymbol{\alpha}_p^{(l)}\left(\boldsymbol{z}_t^{(l+1)}, \boldsymbol{d}_{t-1}^{(l)}\right)\right) \tag{82}$$

$$p\left(\boldsymbol{x}_t|\boldsymbol{z}_{\mathcal{R}_t^{(1)}}^{(1:L)}\right) = \mathcal{D}\left(\boldsymbol{\beta}\left(\boldsymbol{z}_{\mathcal{R}_t^{(1)}}^{(1:L)}\right)\right) \ . \tag{83}$$

The decoder $\boldsymbol{\beta}(\boldsymbol{z}_{\mathcal{R}_t^{(1)}}^{(1:L)})$ is also a WaveNet.

8.2.5 *Clockwork variational autoencoder (CW-VAE)*

The CW-VAE [238] is a hierarchical latent variable model recently introduced for video generation. As illustrated in figures 17 and 18, it is autoregressive in the latent space but not in the observed space, contrary to the VRNN, SRNN and STCN. Additionally, each latent variable is updated only every s_l timesteps, where s_l is a layer-dependent integer, or stride, and $s_1 < s_2 < \cdots < s_L$. This imposes the inductive bias that latent variables exist at different temporal resolutions with $\boldsymbol{z}^{(l)}$ changing at lower frequency than $\boldsymbol{z}^{(l-1)}$. In speech, phonetic variation between $10-400$ ms, morphological and semantic features at the word level and speaker-related variation at the global scale make this a reasonable assumption.

The timesteps at which a layer updates its latent state are given by $\mathcal{T}_l := \{t \in \{1,\dots,T\} \mid t \mod s_l = 1\}$. To simplify temporal notation, we equivalently define this by copying references to the unique states over time, $\boldsymbol{z}_t^{(l)} := \boldsymbol{z}_u^{(l)}$ with $u = \max_\tau\{\tau \in \mathcal{T}_l \mid \tau \leqslant t\}$. The joint distribution factorizes over time and the hierarchy.

$$p(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}^{(1:L)}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\boldsymbol{z}_t^{(1)}) \prod_{l=1}^{L} p(\boldsymbol{z}_t^{(l)}|\boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}) \ .$$

The inference model is conditioned on a window of the observed variable $\boldsymbol{x}_{t:t+s_l}$ depending on the layer's stride s_l.

$$q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = \prod_{l=1}^{L}\prod_{t=1}^{T} q\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}, \boldsymbol{x}_{t:t+s_l}\right) \ .$$

The dependency on $\boldsymbol{x}_{t:t+s_l}$ is encoded via a convolutional ladder network similar to the STCN with $\boldsymbol{d}_t^{(l)} = e(\boldsymbol{x}_{t:t+s_l})$. We define $\boldsymbol{x}_{s_t^l} := \boldsymbol{x}_{t:t+s_l}$ for compactness. The latent state transitions are densely connected and the decoder is also a convolutional network.

$$q\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{x}_{s_t^l}, \boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}\right) = \mathcal{N}\left(\boldsymbol{\alpha}_q^l\left(\boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}, \boldsymbol{d}_t^{(l)}\right)\right)$$

$$p\left(\boldsymbol{z}_t^{(l)}|\boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}\right) = \mathcal{N}\left(\boldsymbol{\alpha}_p^l\left(\boldsymbol{z}_{t-1}^{(l)}, \boldsymbol{z}_t^{(l+1)}\right)\right)$$

$$p\left(\boldsymbol{x}_t|\boldsymbol{z}_{t-s_l/2:t+s_l/2}^{(1)}\right) = \mathcal{D}\left(\boldsymbol{\beta}\left(\boldsymbol{z}_{t-s_l/2:t+s_l/2}^{(1)}\right)\right)$$

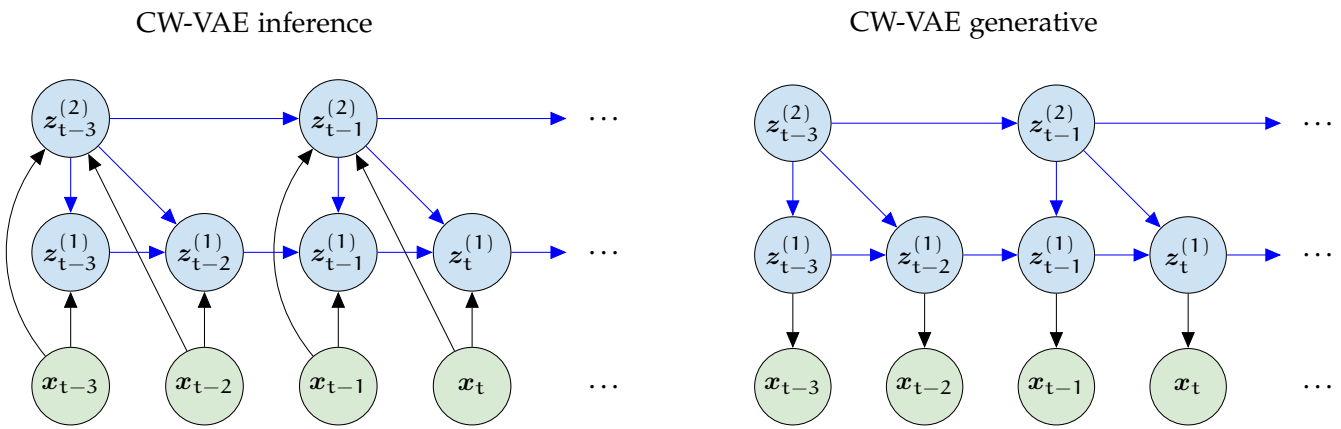


Figure 18: Inference (left) and generative (right) models for the CW-VAE with a hierarchy of $L = 2$ latent variables, $s_1 = 1$ and $s_2 = 2$. The models are unrolled over four consecutive time steps but note that the graph continues towards $t = 0$ and $t = T$. Blue arrows indicate parameter sharing between inference and generative models. We omit the deterministic variable of figure 17 for clarity.

8.2.6 *Speech modeling with Clockwork VAEs*

The video and speech modalities differ in the sampling rates normally used to digitize the natural signals. Sampling rates of common video codecs typically range from 24 Hz up to 60 or 120 Hz. In the speech domain, sampling rates range from 8000 Hz up to e.g. 44 100 Hz commonly used for music recordings. In the original work, s_l is defined as $s_l := k^{l-1}$ for some constant k which makes it exponentially dependent on the layer index l and forces $s_1 = 1$. While this is reasonable for the sample rates of video, training a model at this resolution is infeasible for audio waveform modeling. For this reason, we chose $s_1 \gg 1$ to achieve an initial temporal downsampling and let $s_l := c^{l-1} s_1$ for $l > 1$ and some constant c.

The encoder/decoder networks of the original CW-VAE is not applicable to speech. Hence, we parameterize these networks using 1D convolutions that operate on the raw waveform. We design the encoder as a ladder-network, similar to Aksan and Hilliges [1] and Sønderby et al. [250], as this provides benefits compared to alternatives such as a standalone encoder per latent variable. Specifically, a ladder-network leverages parameter sharing across the latent hierarchy and importantly processes the full observed sequence only once sharing the resulting representations between latent variables. This yields a more computationally efficient encoder and higher activity in latent variables towards the top of the hierarchy.

8.2.7 *Output distribution*

Audio, as well as image data, are naturally continuous signals that are represented in discrete form in computers. The signals are sampled with some bit depth b which defines the range of attainable values, $\boldsymbol{x} \in \{0, 1, \ldots, 2^b - 1\}$. The bit depth typically used in audio and image samplers is between 8 and 32 bit with 8 bit and 16 bit being the most common in the literature (MNIST, [164]; CelebA, [179]; CIFAR10, [156]; TIMIT, [87]; LibriSpeech, [213]).

In the image domain, the discrete nature of the data is usually modeled in one of two ways; either by using discrete distributions [43, 185, 237] or by dequantizing the data and using a continuous distribution [71, 108, 250], which yields a lower bound on the discrete distribution likelihood [259]. In the speech domain, however, the output distribution is often taken to be a continuous Gaussian [114, 162, 289] which was also originally done in VRNN, SRNN and STCN. This choice generally results in an ill-posed problem with a likelihood that is unbounded from above unless the variance is lower bounded [188]. As a result, *reported likelihoods can be sensitive to hyperparameter settings and be hard to compare*. We discuss this phenomenon further in appendix A.8

Most work normalizes the audio or standardizes it to values in a bounded interval $\boldsymbol{x} \in [-1, 1]$. Since $\boldsymbol{x}$ becomes approximately continuous as the bit depth b becomes large and the range of possible values becomes small, this alleviates the issue. However, commonly used datasets with bit-depths of 16 still result in a discretization gap between values that remains much larger than the gap between the almost continuous 32 bit floating point numbers which reinforces the problem [28].

In this work, we therefore benchmark models using a discretized mixture of logistics (DMoL) as output distribution. The DMoL was introduced for use in autoregressive models [237] but has become standard in generative modeling of natural images [43, 185, 264]. It was recently applied to autoregressive speech modeling of raw waveforms [210]. As opposed to e.g. a categorical distribution, the DMoL induces ordinality on the observed space such that values that are numerically close are also considered close in the probabilistic sense. This is a sensible inductive bias for images as well as audio where individual samples represent the amplitude of light or pressure, respectively. We discuss the DMoL for audio further in appendix A.9.

8.3 SPEECH MODELING BENCHMARK

DATA We train models on TIMIT [87], LibriSpeech [213] and Libri-Light [136]. For TIMIT, we randomly sample 5% of the training split for validation. We represent the audio as μ-law encoded PCM standardized to values in $[-1, 1]$ with discretization gap of 2^{-b+1}. We use the original bit depth of 16 bits and sample rate of 16 000 Hz. We use this representation both as the input and the reconstruction target. We provide more details on the datasets in appendix A.3 and additional results on linear PCM in appendix A.7.

LIKELIHOOD We report likelihoods in units of bits per frame (bpf) as this yields a more interpretable and comparable likelihood than total likelihood in nats. It also has direct connections with information theory and compression [243, 261]. In units of bits per frame, lower is better. For LVMs, we report the one-sample ELBO. The likelihoods can be seen in tables 13 and 14. We describe how to convert likelihood to bpf in appendix A.6.

MODELS Architecture and training details are sketched below, while the full details are in appendices A.4 and A.5 along with additional results for some alternative model configurations in appendix A.7. We select model configurations that can be trained on GPUs with a maximum of 12GB of RAM and train all models until convergence on the validation set. We use a DMoL with 10 components for the output distribution of all models and model all datasets at their full bit depth

of 16 bits. We train and evaluate models on stacked waveforms similar to previous work [1, 56, 85] with stack sizes of $s = 1$, $s = 64$ and $s = 256$. Hence, every model input x_t is composed of $\tilde{x}_{t:t+s}$ where $\tilde{x}$ are waveform frames. We provide additional results with a Gaussian output distribution in appendix A.7.

We configure the WaveNet baseline as in the original paper using ten layers per block and five blocks. We set the number of channels, $D_c = 96$. We also provide an LSTM baseline [109]. It uses fully connected encoders and decoders as the VRNN and SRNN but has a single deterministic recurrent cell and much fewer parameters than WaveNet. We report on LSTM models with $D_d = 256$ hidden units.

The configuration of the VRNN and SRNN models is similar to the LSTM. For both models we set the latent variable equal in size to the hidden units, $D_z = 256$. At stack size $s = 1$, the models are computationally demanding and hence we train them on randomly sampled segments from each training example and only on TIMIT.

The STCN is used in the "dense" configuration of the original work [1]. It uses 256 convolutional channels and $L = 5$ latent variables of dimensions $16, 32, 64, 128, 256$ from the top down. We also run a one-layered ablation with the same architecture but only one latent variable of dimension 256 at the top. The CW-VAE is configured similar to the VRNN and SRNN models and with $c = 8$. We run the CW-VAE with $L = 1$ and 2 layers of latent variables. The number of convolutional channels and is set equal to D_z which is set to 96.

BASELINES We supply three elementary baselines that form approximate upper and lower bounds on the likelihood for arbitrary s. Specifically, we evaluate an uninformed per-frame discrete uniform distribution and a two-component DMoL fitted to the training set to benchmark worst case performance. We also report the likelihood achieved by the lossless compression algorithm, FLAC [59] which establishes a notion of good performance, although not a strict best case. We report FLAC on linear PCM since it was designed for this encoding.

TIMIT For temporal resolutions of $s = 1$, the deterministic autoregressive models yield the best likelihoods with WaveNet achieving 10.88 bpf on TIMIT as seen in table 13. Somewhat surprisingly, the LSTM baseline almost matches WaveNet with a likelihood of 11.11 bpf at $s = 1$. However, due to being autoregressive in training, the LSTM trains considerably slower than the parallel convolutional WaveNet; something not conveyed by table 13. Notably, the VRNN and SRNN models achieve likelihoods close to that of WaveNet and the LSTM at around 11.09 bpf. The STCN exhibited instability when trained at $s = 1$ and tended to diverge.

s	**Model**	**Configuration**	$\mathcal{L}$ **[bpf]**
1	Uniform	Uninformed	16.00
1	DMoL	Optimal	15.60
-	FLAC	Linear PCM	**8.582**
1	WaveNet	$D_c = 96$	**10.88**
1	LSTM	$D_d = 256$	10.97
1	VRNN	$D_z = 256$	$\leqslant$11.09
1	SRNN	$D_z = 256$	$\leqslant$11.19
1	STCN-dense	$D_z = 256, L = 5$	$\leqslant$11.77
64	WaveNet	$D_c = 96$	13.30
64	LSTM	$D_d = 256$	13.34
64	VRNN	$D_z = 256$	$\leqslant$12.54
64	SRNN	$D_z = 256$	$\leqslant$12.42
64	CW-VAE	$D_z = 96, L = 1$	$\leqslant$12.44
64	CW-VAE	$D_z = 96, L = 2$	$\leqslant$12.17
64	STCN-dense	$D_z = 256, L = 1$	$\leqslant$12.32
64	STCN-dense	$D_z = 256, L = 5$	$\leqslant$**11.78**
256	WaveNet	$D_c = 96$	14.11
256	LSTM	$D_d = 256$	14.20
256	VRNN	$D_z = 256$	$\leqslant$13.27
256	SRNN	$D_z = 256$	$\leqslant$13.14
256	CW-VAE	$D_z = 96, L = 1$	$\leqslant$13.11
256	CW-VAE	$D_z = 96, L = 2$	$\leqslant$12.97
256	STCN-dense	$D_z = 256, L = 1$	$\leqslant$13.07
256	STCN-dense	$D_z = 256, L = 5$	$\leqslant$**12.52**

Table 13: Model likelihoods $\mathcal{L}$ on TIMIT represented as a 16bit μ-law encoded PCM for different stochastic latent variable models compared to deterministic autoregressive baselines. For the CW-VAE, s refers to s_1 and the multi-layered models have $c = 8$. Likelihoods are given in units of bits per frame (bpf).

At $s = 64$, Wavenet and the LSTM yield significantly worse likelihoods than all LVMs separated by ~ 1 bit. The CW-VAE outperforms the VRNN and SRNN when configured with a hierarchy of latent variables. With a single layer of latent variables, the CW-VAE is inferior to both SRNN and VRNN but notably still better than the LSTM. These observations carry to $s = 256$, where a multilayered CW-VAE outperforms the LSTM, VRNN and SRNN. The STCN yields the best results at both $s = 64$ and $s = 256$. For strides $s > 1$, previous work has attributed the inferior performance of autoregressive models without latent variables, such as WaveNet and the LSTM, to the ability of LVMs to model intra-step correlations [161].

Decreasing the resolution s improves the likelihood for all LVMs. However, the best performing models, STCN and CW-VAE are not yet scalable to this regime for reasons related to numerical instability and computational infeasibility. This seems to indicate that LVMs may be able to outperform autoregressive models at $s = 1$ in the future.

LIBRISPEECH On LibriSpeech (table 14), results are similar to TIMIT. The STCN achieves the best likelihood at $s = 64$ and the CW-VAE surpasses WaveNet and the LSTM.

COMPRESSION A connection can be made between the model likelihoods and the compression rates of audio compression algorithms. Lossy compression algorithms, such as MP3, exploit the dynamic range of human hearing to achieve 70-95% reduction in bit rate [34] while lossless compression algorithms, such as FLAC, achieve 50-70% reduction [59] independent of audio content. Although both the autoregressive models and the LVMs are lossy, their objective minimizes the amount of incurred loss. The best likelihoods reported in tables 13 and 14 correspond to about a 30% reduction in bit rate which indicates that there are significant gains in likelihood to be made in speech modeling.

8.3.1 *Likelihood results*

8.4 PHONEME RECOGNITION

Although the likelihood is a practical metric for model comparison and selection, a high likelihood does not guarantee that a model has learned useful representations [116]. For speech data, we would expect models to learn features related to phonetics which would make them useful for tasks such as automatic speech recognition (ASR). The Mel spectrogram is a well-established representation of audio designed for speech recognition and is predefined rather than learned. To assess the usefulness of the representations learned by the benchmarked models, we compare them to the highly useful Mel spectro-

s	**Model**	**Configuration**	**Likelihood $\mathcal{L}$ [bpf]**			
			dev-clean 10h/100h	dev-other 10h/100h	test-clean 10h/100h	test-other 10h/100h
1	Uniform	Uninformed	16.00	16.00	16.00	16.00
1	DMoL	Optimal	15.66	15.70	15.62	15.71
-	FLAC	Linear PCM	**9.390**	**9.292**	**9.700**	**9.272**
1	Wavenet	$D_c = 96$	**10.96/10.89**	**10.85/10.76**	**11.12/11.01**	**11.05/10.85**
1	LSTM	$D_d = 256$	11.21/11.17	11.10/11.06	11.35/11.29	11.28/11.23
64	Wavenet	$D_c = 96$	13.61/13.24	13.58/13.21	13.61/13.22	13.60/13.21
64	LSTM	$D_d = 256$	13.56/13.25	13.52/13.24	13.55/13.23	13.56/13.25
64	CW-VAE	$D_z = 96, L = 1$	$\leqslant$12.32/12.24	12.32/12.23	12.43/12.33	12.43/12.33
64	CW-VAE	$D_z = 96, L = 2$	$\leqslant$12.30/12.22	12.30/12.21	12.40/12.31	12.39/12.32
64	STCN-dense	$D_z = 256, L = 5$	$\leqslant$**11.98/11.90**	**11.98/11.89**	**12.08/12.02**	**12.09/12.01**

Table 14: Model likelihoods $\mathcal{L}$ on LibriSpeech test sets represented as 16 bit μ-law encoded PCM. For the CW-VAE, s refers to s_1 and the two-layered models have $s_2 = 8s_1$. The models are trained on either the 10 h LibriLight subset or the 100 h LibriSpeech train-clean-100 subset as indicated. Likelihoods are given in units of bits per frame (bpf).

ASR configuration			Result
Data	Model	Input	PER [%]
3.7h	Spectrogram	Mel	24.1
3.7h	WaveNet	$\boldsymbol{h}^{(15)}$	27.7
3.7h	LSTM	$\boldsymbol{h}$	23.0
3.7h	VRNN	$\boldsymbol{z}$	23.2
3.7h	SRNN	$\boldsymbol{z}$	26.0
3.7h	CW-VAE	$\boldsymbol{z}^{(1)}$	36.4
3.7h	STCN	$\boldsymbol{z}^{(2)}$	**21.9**
1.0h	Spectrogram	Mel	30.8
1.0h	WaveNet	$\boldsymbol{h}^{(15)}$	34.7
1.0h	LSTM	$\boldsymbol{h}$	30.1
1.0h	VRNN	$\boldsymbol{z}$	30.4
1.0h	SRNN	$\boldsymbol{z}$	31.7
1.0h	CW-VAE	$\boldsymbol{z}^{(1)}$	40.0
1.0h	STCN	$\boldsymbol{z}^{(2)}$	**26.7**
10m	Spectrogram	Mel	47.1
10m	WaveNet	$\boldsymbol{h}^{(15)}$	52.8
10m	LSTM	$\boldsymbol{h}$	46.1
10m	VRNN	$\boldsymbol{z}$	44.6
10m	SRNN	$\boldsymbol{z}$	47.3
10m	CW-VAE	$\boldsymbol{z}^{(1)}$	54.9
10m	STCN	$\boldsymbol{z}^{(2)}$	**42.7**

Table 15: Evaluation of learned representations via phoneme recognition on TIMIT. The ASR model is a three-layered bidirectional LSTM trained with CTC [93]. The experiment is similar to that of Hsu, Zhang, and Glass [114] but we focus on the effect of the amount of labeled data and evaluate many more models. The specific representations used is indicated in the input column.

gram on the task of phoneme recognition. Phonemes are fundamental units of speech that relate to how parts of words are pronounced rather than characters or words themselves (see also appendix A.12).

QUANTITATIVE We train an ASR model to recognize phonemes and compare its performance when using input representations obtained from different unsupervised models. For WaveNet and the LSTM, we use the hidden state as the representation. For all LVMs, we use the latent variable. For the hierarchical LVMs and WaveNet, we run the experiment using each possible representation and report only the best one here. We compare the learned representations to a log Mel spectrogram with 80 filterbanks, hop length 64 and window size 128. We also compared to using raw PCM in vectors of 64 elements standardized to $[-1, 1]$ but found that the ASR did not reliably converge at all which highlights the importance of input representation. The ASR model is a three-layered bidirectional LSTM with 256 hidden units. It is trained with the connectionist temporal classification (CTC) loss [93] which lets it jointly learn to align and classify without using label alignments. We pre-train all unsupervised models at $s = 64$ on the full TIMIT training dataset excluding the validation data (3.7h) as in table 13. We then train the ASR model on all 3.7h as well as 1h and 10m subsets. We report results on the test set in terms of phoneme error rates (PER) in table 15.

As expected, Mel spectrogram performs well achieving 24.1% PER using 3.7 hours of labeled data. However, the ASR trained on STCN representations outperforms the Mel spectrogram with a PER of 21.9%. This indicates that unsupervised STCN representations are phonetically rich and potentially better suited for speech modeling than the engineered Mel spectrogram. When the amount of labeled data is reduced, LVM representations suffer slightly less than deterministic ones. WaveNet representations are interestingly outperformed by both the LSTM and all LVMs.

QUALITATIVE We qualitatively asses the learned latent representations selectively for the CW-VAE. We infer the latent variables of all utterances by a single speaker from the TIMIT test set. We sample the latent sequence 100 times to estimate the mean representation per time step. We then compute the average latent representation over the duration of each phoneme using aligned phoneme labels. This approximately marginalizes out variation during the phoneme. We use linear discriminant analysis (LDA) [84] to obtain a low-dimensional linear subspace of the latent space. We visualize the resulting representations colored according to test set phoneme classes in figure 19. We note that many phonemes are separable in the linear subspace and that related phonemes such as “s" and “sh" are close.

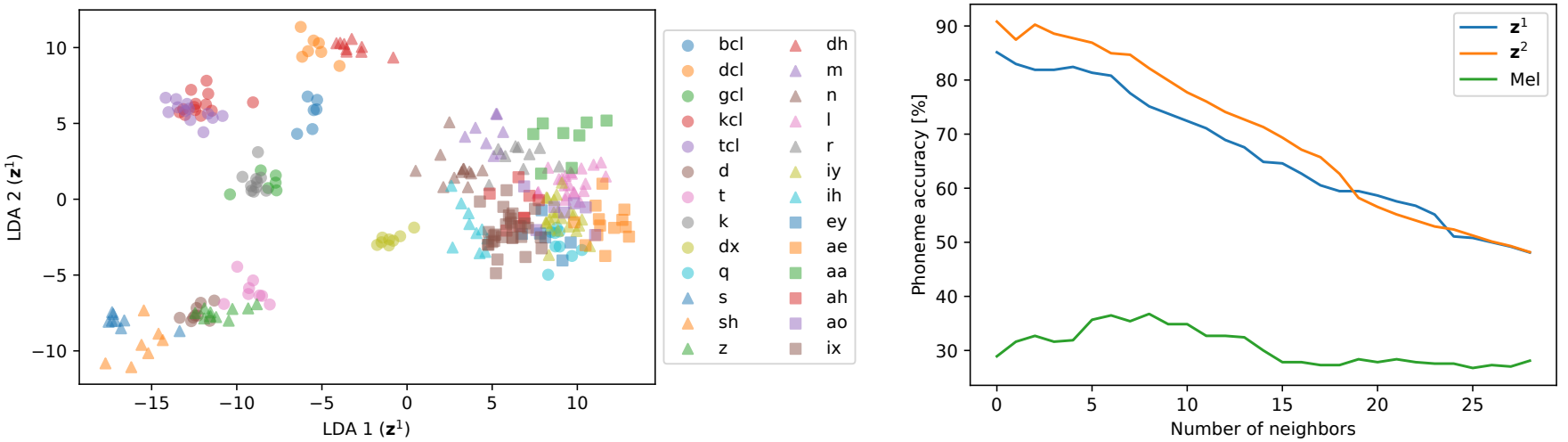


Figure 19: (left) Clustering of phonemes in a 2D Linear Discriminant Analysis (LDA) subspace of a CW-VAE latent space ($z^{(1)}$). (right) Leave-one-out phoneme classification accuracy for a KNN classifier at different K in a 5D LDA subspace of a CW-VAE latent space.

We also show the average accuracy of a leave-one-out k-nearest-neighbor (KNN) classifier on the single left-out latent representation reduced with a 5-dimensional LDA as a function of the number of neighbors. We compare accuracy to a Mel-spectrogram averaged over each phoneme duration and LDA reduced. The spectrogram is computed with hop length set to 64, equal to s_1 for the CW-VAE, window size 256 and 80 Mel bins. We see that both latent spaces yield significantly better KNN accuracies than the Mel features.

8.5 CONCLUSION

In this paper, we developed a benchmark for speech modeling with stochastic latent variable models (LVMs). We compared LVMs and deterministic autoregressive baseline models on equal footing and find that LVMs achieve inferior likelihood compared to deterministic WaveNet and LSTM baselines. Surprisingly, the LSTM almost matches the popular WaveNet model. We see that hierarchical LVMs, such as STCN and CW-VAE, outperform non-hierarchical versions of themselves in ablation experiments as well as non-hierarchical LVMs such as VRNN and SRNN. This matches recent observations in the image domain. We note that the STCN with an autoregressive decoder outperforms the non-autoregressive CW-VAE, which we adapted to speech. Finally, we find that LVMs can learn latent representations that are useful for phoneme recognition and even surpass Mel spectrograms, which are tailored for the task, when identical models are trained on top of the representations. While the best performing models are not yet scalable to the highest temporal resolution, these results indicate that they are able to improve upon deterministic models in the future.

9

DO WE STILL NEED AUTOMATIC SPEECH RECOGNITION FOR SPOKEN LANGUAGE UNDERSTANDING?

ABSTRACT

Spoken language understanding (SLU) tasks are often solved by first transcribing an utterance with automatic speech recognition (ASR) and then feeding the output to a text-based downstream model. Recent advances in self-supervised representation learning for speech have focused on improving the ASR component. We investigate whether representation learning for speech has matured enough to replace ASR in SLU. We compare learned speech features from wav2vec 2.0, state-of-the-art ASR transcripts, and the ground truth text as input for four SLU tasks: A novel speech-based named entity recognition task, a cardiac arrest detection task on real-world emergency calls, intent classification, and speech translation. We show that learned speech features are competitive with ASR transcripts on three classification tasks. In most cases, they are the better choice. For speech translation, ASR transcripts are still superior. We highlight the intrinsic robustness of wav2vec 2.0 representations to out-of-vocabulary words as key to better performance.

9.1 INTRODUCTION

Pre-training large transformers with self-supervised learning (SSL) has dramatically advanced automatic speech recognition (ASR) [13] and shown promise for taking spoken language understanding (SLU) to the next level [159, 280]. However, it remains an open question whether SSL is ready to replace ASR in the SLU pipeline. Labeled data for ASR training is often difficult to obtain, so if self-supervised representations offer a competitive alternative, it has the potential to democratize SLU.

Work on speech representation learning has primarily focused on ASR [13] and other speech-specific tasks, such as emotion recognition [217], speaker identification [176] and phoneme classification [10]. Tasks that require rely on semantic features have been studied less [159, 280]. Accordingly, the number of SLU tasks is limited, and many text-based natural language understanding tasks cannot be directly translated to the speech domain due to the difficulty of obtaining word segmentation. For this reason, existing tasks only contain little data [181] or make use of synthetic speech [280].

In this work, we compare speech features from the wav2vec 2.0 framework, state-of-the-art ASR transcripts, and ground truth text as input for four SLU tasks that all require knowledge about semantic concepts. While the quality of an ASR model can always be debated, the ground truth text defines an upper bound on ASR performance and serves as a baseline, when available. We consider existing intent classification (IC) and speech translation (ST) tasks and present a novel speech-based named entity recognition (NER) task. Finally, we use a proprietary dataset of 911-calls to define a noisy real-world task of cardiac arrest detection (CAD). Our contributions are as follows:

1. We present the first systematic comparison between text and speech features as input to a broad range of SLU tasks since the recent surge in SSL for speech.
2. We show that wav2vec 2.0 speech representations are better than ASR transcripts for downstream IC, NER and CAD tasks when labeled data is limited and competitive when data is abundant. For MT, the speech representations fall short of text.
3. We introduce a speech-based NER task derived from the most widely used dataset for research in ASR and SSL for speech, LibriSpeech.

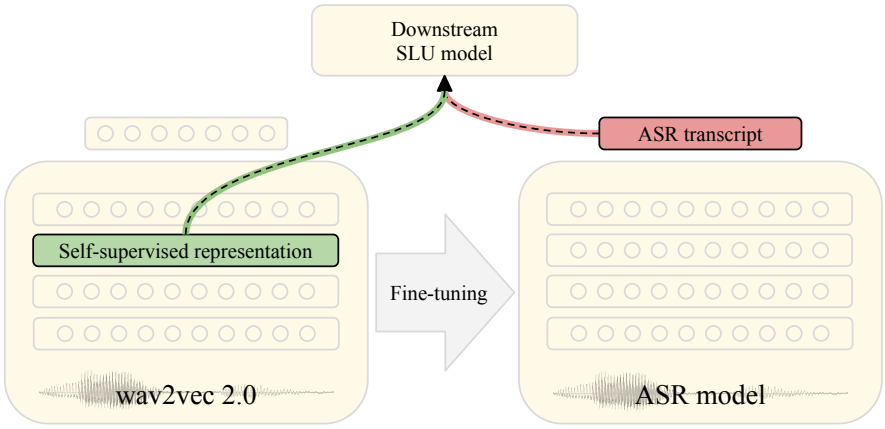


Figure 20: Self-supervised models, such as wav2vec 2.0, yield state-of-the-art results when fine-tuned for automatic speech recognition. However, we show this step is redundant for many downstream spoken language understanding tasks where self-supervised representations can be used as input.

9.2 TASKS

We start by presenting the four tasks, all based on natural speech. Dataset statistics are found in table 16.

9.2.1 *Named entity recognition: LibriSpeech*

LibriSpeech [213] is derived from audiobooks part of the LibriVox project[1]. Training data for wav2vec 2.0 consist of 60K hours of speech from LibriVox, while the open-source ASR models used in this work are trained on LibriSpeech unless stated otherwise. Defining a downstream task on data from the same domain used to train the SSL model and ASR model corresponds to a common scenario where training data for the different modeling steps overlap. LibriSpeech comes with multiple standardized training subsets [136] allowing us to study how the downstream model is affected by varying the amount of training data. Finally, LibriSpeech contains two validation and test subsets, *clean* and *other*, which offer insight into the importance of recording quality.

We provide silver label named entity tags for LibriSpeech. The labels were obtained by using an off-the-shelf Electra language model[2] fine-tuned on the CoNLL-2003 NER task [58, 275] and applying the model to the ground truth text. We manually reviewed model-induced labels for the validation set to get a sense of data quality. For 1,000 randomly selected examples, human-model agreement is high, with a Krippendorff's alpha of 0.98.

The task is to predict whether a named entity is contained in the input example. In contrast to classic text-based NER, where each word is tagged with a label, we considered this a binary sequence-based classification task to keep the model setup for text and speech features as similar as possible.

Conveniently, we find that the dataset is balanced such that approximately 50% of the examples in the training subsets contain a named entity. For validation and test, the fraction is around 30%. We make the dataset available with more details[3], such as entity type (i.e., *person*, *location*, *organization* or *miscellaneous*) and entity alignment obtained from [181] which used the Montreal Forced Aligner[4].

9.2.2 *Cardiac arrest detection: Seattle Fire Department*

From a proprietary dataset of 911 emergency calls provided by Seattle Fire Department, WA, USA, we constructed a binary sequence

1 https://librivox.org
2 https://huggingface.co/dbmdz
3 https://github.com/borgholt/ner-librispeech
4 https://montreal-forced-aligner.readthedocs.io

Task - Dataset	Number of examples			Duration	Median	WER
	train	valid	test	[h]	[s]	[%]
Named entity recognition - LibriSpeech	281,241	5,567	5,559	982	16.7	2.1 / 4.5
Cardiac arrest detection - Seattle Fire Department	2,086	260	260	146	171.5	35.1
Intent classification - Fluent Speech Commands	23,132	3,118	3,793	19	1.8	23.2
Speech translation - CoVoST 2 (En-De)	289,165	15,517	15,524	478	5.4	16.6

Table 16: Basic dataset statistics for the SLU tasks. *Number of examples* in the subsets. *Duration* in hours for all subsets. *Median* example length in seconds. Word error rate (WER) on the validation set for the ASR models presented in 9.3.2.

classification task where the objective is to predict whether the caller describes an out-of-hospital cardiac arrest (OHCA) or not. The original dataset contains 1303 OHCA calls and many more not-OHCA calls. We did a random 80-10-10 split of the OHCA calls and sampled a not-OHCA call of similar length to each of the OHCA calls to keep the dataset balanced in terms of target distribution and hours of speech per class. We did not have ground truth text available for this task but report word error rate on a separate subset in table 16.

9.2.3 *Intent classification: Fluent Speech Commands*

The Fluent Speech Commands (FSC) dataset [181] consists of 248 uniq-ue read-speech commands from 97 speakers instructing a hypothetical intelligent home device to perform an action (e.g., "Turn the lights on in the kitchen"). Recording of the commands was crowd-sourced, resulting in a varied selection of English speakers from the US and Canada. The task was originally phrased as a multi-slot task with three slots: *action*, *object* and *location*. However, due to the small number of slot classes, the task is commonly rephrased as a simple classification task with 31 unique classes.

The original training, validation, and test subsets contain the same commands spoken by different speakers. With this setup, the task essentially amounts to sentence-level speech recognition. Thus, we also consider an alternative split proposed by Arora et al. [5]. The so-called *challenge* split contains two test sets: A *speaker* subset and an *utterance* subset. The first contains particularly challenging speakers. The latter contains commands that have been excluded from the training set, which is very useful when probing for semantic features.

9.2.4 *Speech translation: CoVoST 2*

CoVoST 2 is a multilingual speech-to-text translation dataset [269] derived from the Common Voice speech corpus [4]. Translations were made by professional translators and the corresponding speech recordings were crowd-sourced. We focused on the English-to-German task using the so-called CoVoST training set and the Common Voice test and validation sets as in the original work.

9.3 EXPERIMENTS

9.3.1 *Task-specific models*

We are interested in comparing the information content of the *input representations*, not the models, so we chose a minimalist architecture. All models take as input a sequence of vectors $\boldsymbol{x}_{1:T} = \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T$ where $\boldsymbol{x}_t \in \mathbb{R}^K$ and share a similar encoder. A fully connected layer

without activation maps each x_t to a D-dimensional linear subspace. This linear mapping is the only source of variation in terms of model parameterization between the input representations as it depends on the input dimensionality K: 1024 for wav2vec 2.0 representations, 29 for character-level text, and 1,296 to 41,341 for word-level text depending on vocabulary size which we treated as a hyperparameter. The linearly projected features are fed to a bidirectional LSTM with a D-dimensional recurrent state.

Hereafter, each task requires a different architecture. For the binary NER and CAD tasks, the LSTM output $h_{1:T}$ is max-pooled and fed into a single neuron with a sigmoid activation to parameterize a Bernoulli distribution. Similarly, for the IC task, the LSTM output is pooled and mapped to a 31-dimensional vector with softmax normalization to parameterize a categorical distribution. For the MT task, we used an LSTM-based autoregressive decoder with scaled dot-product attention [266]. We used a vocabulary of 10K subword units[5] for the target language.

9.3.2 *Speech representations and ASR models*

The wav2vec 2.0 models use contrastive self-supervised learning and can be fine-tuned for ASR with a connectionist temporal classification loss. See [13] for more details. We considered two SSL-ASR model pairs downloaded from the FAIRSEQ sequence modeling toolkit[6]. For the first pair, the self-supervised wav2vec 2.0 model has been trained on 60K hours of speech from LibriLight and fine-tuned on 960 hours from LibriSpeech [13]. The second pair, which is more robust, adds 3000 hours of conversational and crowd-sourced speech from the Fisher, Switchboard and CommonVoice corpora to the self-supervised training, while the ASR model was fine-tuned using the 300 hours from Switchboard [112]. All models use the same architecture. In order to choose which pair to use for a given task, we tested the two ASR models on the validation set for each task and chose the model pair corresponding to the lowest word error rate. For the IC and CAD tasks, the robust ASR model was better.

As shown in [10], the top layers of wav2vec 2.0 are a poor choice of input to phoneme classification. We ran a small initial experiment with limited training data to determine which output from the 24 transformer layers in the wav2vec 2.0 architecture to use as input to the downstream tasks. We found that layer 15 yielded the best results. This layer has also been found to provide the best results for phoneme classification [10], and layers 13 through 16 have been shown to be the most similar with text-based word embeddings [215].

5 https://github.com/google/sentencepiece
6 https://github.com/pytorch/fairseq

Input	Source	10 hours		100 hours		960 hours	
		clean	other	clean	other	clean	other
Characters	GT	57.5±1.6	60.2±1.9	78.4±0.9	81.4±0.7	90.9±0.3	91.4±0.3
Word	GT	55.9±1.6	58.1±1.5	74.1±0.7	76.4±0.8	89.2±0.4	90.6±0.6
Characters	ASR	55.4±3.1	60.4±1.9	77.6±0.9	79.5±0.9	**89.8±0.4**	**88.1±0.4**
Word	ASR	54.3±1.3	56.7±1.6	73.7±0.9	75.4±1.1	87.4±0.6	86.3±0.5
Wav2vec 2.0	SSL	**83.4±0.6**	**81.4±0.6**	**87.7±0.4**	**84.5±0.4**	88.7±0.5	85.4±0.4
Characters	$\text{ASR}_{10\text{min}}$	55.8±2.0	56.1±2.5	68.3±0.9	69.5±0.8	82.3±0.5	80.5±0.4
Characters	$\text{ASR}_{100\text{h}}$	56.8±1.3	60.6±2.0	77.1±0.6	78.0±0.8	88.5±0.3	87.1±0.4

Table 17: Named entity recognition F1-scores on the LibriSpeech test sets.

9.3.3 *Training*

All models were trained by minimizing cross-entropy and use $D = 256$. In the very low-resource settings, we also tested smaller dimensionalities to reduce overfitting, but this did not improve results. We used the Adam optimizer [146] with a fixed learning rate of $3 \cdot 10^{-4}$ for the first half of training before annealing it to $5 \cdot 10^{-5}$ during the second half. Batch size and validation frequency were tuned for each task on the ASR character-level. We ensured that the number of training steps was large enough to reach convergence for all experiments.

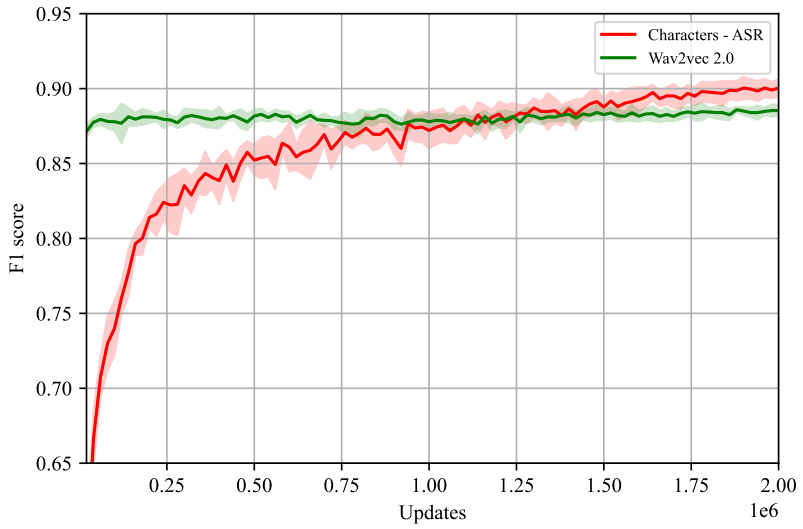


Figure 21: F1-score on the clean validation subset of LibriSpeech during training with 960 hours of data for the downstream task. As in the table, figure shows results meaned over 10 runs with standard deviation. The model trained on wav2vec 2.0 representations converged very fast, which suggests that they encode useful semantic features.

Input	Source	CAD F1-score	IC Accuracy	ST BLEU
Characters	GT	N/A	100.0 ± 0	15.3 ± 0.3
Word	GT	N/A	100.0 ± 0	13.8 ± 0.5
Characters	ASR	83.3 ± 2.0	99.4 ± 0.1	$\mathbf{11.4 \pm 0.2}$
Word	ASR	80.4 ± 2.1	98.6 ± 0.1	10.8 ± 0.3
Wav2vec 2.0	SSL	83.0 ± 1.2	$\mathbf{99.5 \pm 0.0}$	6.1 ± 0.2

Table 18: Results for cardiac arrest detection, intent classification, and speech translation. Bold indicates a significant difference between best ASR-based result and wav2vec 2.0 representations.

9.4 RESULTS

Results for each of the four tasks are presented below. We report the metric commonly used in previous work for the existing tasks. *GT* refers to *ground truth* in tables 2 and 3. All results are given as a mean and standard deviation over 10 runs. Boldfaced numbers indicate a significant difference ($p < 0.05$) between the best ASR-based results and the wav2vec 2.0 results, which was tested with a Mann-Whitney U test.

9.4.1 *Named entity recognition: LibriSpeech*

The wav2vec 2.0 representations showed impressive performance on the 10-hour subset, as seen in table 17, where text-based models were only slightly better than a random baseline. Even with 100 hours of labeled data, they were superior. The gap closed at 960 hours, where character-level ASR transcripts were slightly better.

Generally, we would expect performance to degrade with the quality of the speech recognition model. But faced with an overlap between ASR and NER training data, it might not be the case. The ASR might overfit, which will result in a mismatch between errors on training and evaluation data. The two bottom rows of table 17 show results with limited ASR training data. Overall, performance is degraded by using less than the full 960 hour. However, with only 100 hours of training data, the performance drop is fairly small.

In general, the models trained with wav2vec 2.0 representations converged much faster than the text-based models. Figure 21 shows the development of the F1-score on the clean validation subset for the models trained on 960 hours. The same trend was observed for the other NER experiments, which suggests that wav2vec 2.0 representations encode semantic features not easily learned from text.

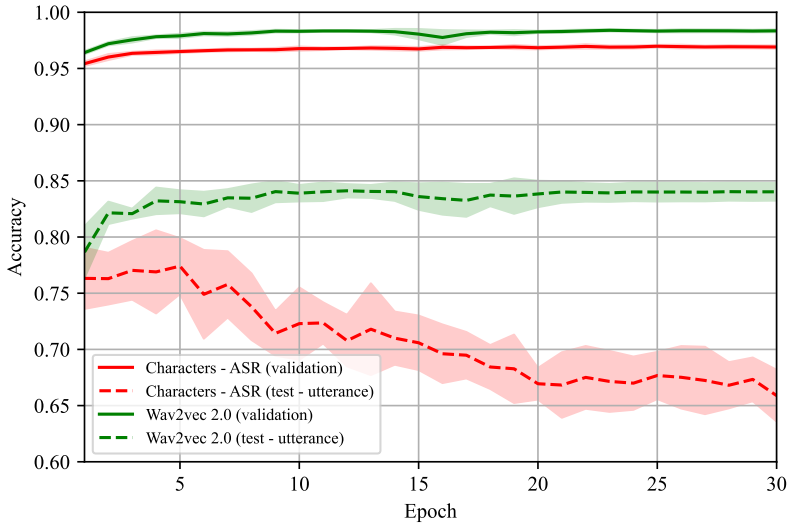


Figure 22: Accuracy on the validation and utterance test set of the challenge split from [5]. As training progresses, the model trained on wav2vec 2.0 representations generalize to both validation and test set. The ASR-based model does not generalize to the test set.

Models trained on text performed better on the *other* subset, whereas models trained in speech representations always performed best on the *clean* subset, highlighting the speech features' sensitivity to noisy conditions. Although the ASR transcripts were also affected by noise, they gave more robust results, as these models performed better on the *other* subset in all but one case.

On examples that exclusively contain named entities that are out-of-vocabulary, wav2vec 2.0 representations gave an error rate of 23% when trained on 100 hours. ASR transcripts gave a substantially higher error rate of 36%. This underscores the large amount of data needed for robust out-of-vocabulary named entity recognition.

9.4.2 *Cardiac arrest detection: Seattle Fire Department*

Considering the observation that ASR transcripts are more noise-robust than wav2vec 2.0 representations, we might expect them to fare better on noisy 911-calls. However, as seen in table 18, the wav2vec 2.0 representations still yielded competitive results. The difference between transcripts and wav2vec 2.0 representations were not significant ($P - value = 0.12$). Unlike the NER task, it is possible that speech-based features, such as emotion and rate of speech, might prove useful for this task.

Input	Source	IC - challenge split	
		Speaker	Utterance
Characters	GT	100.0 ± 0	81.7 ± 2.4
Word	GT	100.0 ± 0	75.2 ± 2.6
Characters	ASR	93.48 ± 0.3	69.5 ± 3.2
Word	ASR	86.7 ± 0.2	70.9 ± 3.7
Wav2vec 2.0	SSL	$\mathbf{97.6 \pm 0.1}$	$\mathbf{84.1 \pm 0.7}$

Table 19: Intent classification accuracies on the *challenge* split proposed in [5]. Bold indicates a significant difference between best ASR-based result and wav2vec 2.0 representations.

9.4.3 *Intent classification: Fluent Speech Commands*

As mentioned in the task description, every speaker in this dataset read the same 248 commands. As a result, training, validation and test subsets contain the same 248 identical examples when we consider ground truth text, which leads to an accuracy of 100% as seen in table 18. The wav2vec 2.0 representations were slightly better than the ASR transcripts and close to the more complex state-of-the-art ASR-based system from [225] which reached 99.7% accuracy.

On the more interesting challenge split, we found a more pronounced difference between ASR transcripts and wav2vec 2.0 representations as seen in table 19. On the utterance test set, which contains commands not included in the training data, wav2vec 2.0 representations were much better. The ground truth text also yielded better results than ASR transcripts, but not on par with wav2vec 2.0 representations. In addition, the wav2vec 2.0 results presented here, are not far off models trained on ground truth text with FastText (83.7%) and BERT (86.1%) embeddings from [5]. The speaker test set showed the same picture as with the standard split, although performance was somewhat degraded.

Because the validation set commands of the challenge split also overlap with the training set, we found it relevant to compare the development of validation set and utterance test set during training post hoc. The results are plotted in figure 22. Interestingly, while the learnings of both models generalize to the validation set, only the model trained on wav2vec 2.0 representations improved on the utterance test set as well. Again, this suggests that the wav2vec representations encode features useful for spoken language understanding.

9.4.4 *Speech translation: CoVoST 2*

Unsurprisingly, we find that our simple ASR-based ST system was a lot worse than ground truth text. The wav2vec 2.0 representations were even worse. These results are not surprising considering the generally large gap between speech and text-based approaches [269]. We hypothesize that the lack of simple morphological features, like word boundaries, is a challenge to overcome for a shallow model trained on speech-based representations. To test this hypothesis, we trained a model on the ASR character-level transcripts without white-spaces (e.g., HOW ARE YOU $\rightarrow$ HOWAREYOU) which resulted in a notable drop from BLEU 11.5 ± 0.2 to 9.7, but not enough to explain the gap between the two representations.

9.5 DISCUSSION

This work should not be seen as a quest to remove ASR from the SLU pipeline. Automatically generated transcripts offer an important layer of interpretability in modern speech applications. And as we saw, there are tasks for which transcripts are still the better option. Furthermore, we did not explore how text-based language models can be modified to handle error-prone transcripts, which is a promising direction for SLU [159]. However, this work is highly relevant when large quantities of unlabeled speech data can be easily obtained, but no or limited text data is readily available – such as in an emergency call center. Future work may explore how to directly fine-tune representation models like wav2vec 2.0 for SLU tasks. This approach has been very successful for speech recognition (e.g., the ASR models used in this study).

9.6 CONCLUSION

We compared self-supervised speech features from wav2vec 2.0 with automatic speech recognition transcripts and ground truth text as input to a simple model on four spoken language understanding tasks. Interestingly, wav2vec 2.0 representations proved highly competitive for cardiac arrest detection, named entity recognition, and intent classification. In most cases, the speech representations outperformed the transcripts. Only when 960 hours of labeled training data was available, the transcripts yielded a slight improvement on the named entity recognition task. For speech translation, the wav2vec 2.0 representations were inferior to the text-based features. By analyzing training and error patterns, we found evidence that suggests that wav2vec 2.0 representations encode semantic features directly useful for spoken language understanding tasks. Our results on the classification tasks have implications for how to tackle spoken language understanding

tasks with limited training data demonstrating that the traditional automatic speech recognition step can be bypassed.

Part IV

DISCUSSION AND CONCLUSIONS

10 DISCUSSION

The work presented in the previous six chapters has been carried out over the last three years. During that period, speech processing has evolved greatly. The same goes for the author's knowledge and perception of the problems treated in each of the chapters. In addition, each study is aimed at publication at a venue that pose limitations on the extent of the written presentation. Thus, selected chapters will receive special treatment and be discussed further below. Chapter 11 will conclude on all chapters.

10.1 CHAPTER 4 REVISITED: THE CASE FOR CONTEXT

In chapter 4, end-to-end speech recognition models was compared in terms of their ability to utilize context. The straight-forward conclusion was that the models rely on context, and not just on the speech segment corresponding to a word.

10.1.1 *End-to-end speech recognition models use context, but ...*

Although this may not be surprising, there is plenty of reason to believe that these models could in fact be better at utilizing context. For example, data augmentation by masking consecutive time steps of the input, and thereby forcing the model to infer the masked portion from context, has become a mainstay in end-to-end speech recognition [214]. Furthermore, language model decoding for end-to-end speech recognition is motivated by the same idea. To see this, let us review the simple expression[1] we try to maximize when decoding with a language model,

$$\mathrm{P}(\mathbf{w}|\mathbf{x})\mathrm{P}(\mathbf{w}) \ . \tag{84}$$

Here, $\mathbf{x}$ is an acoustic input and $\mathbf{w} = (w_1, w_2, \ldots, w_T)$ is a sequence of words. $\mathrm{P}(\mathbf{w}|\mathbf{x})$ is given by an end-to-end speech recognition model and $\mathrm{P}(\mathbf{w})$ is given by an autoregressive language model. The models used in chapter 4 output characters, not words, but we will use words here to ease discussion. The language model probability factorizes according to the chain rule of probability, so we have

$$\mathrm{P}(\mathbf{w}) = \mathrm{P}(w_1)\mathrm{P}(w_2|w_1)\ldots\mathrm{P}(w_T|w_{<T}) \ . \tag{85}$$

1 In practice, this is too constraining. It is common to down-weigh the language model and use an insertion penalty [99].

For the expression in equation 84 to be maximized, the probability of each word w_t should be high given its preceding context. Of course, external language models are often trained on more data (in terms of the number of words) than used for training an end-to-end speech recognition model which requires labeled data. Thus, if a speech recognition model improves with language model decoding, it is not necessarily a sign that the model is not capable of utilizing context. Rather, it just shows that the language model is better at this, which is also to be expected given more training examples and a tailored training objective (i.e., $P(w_t|w_{<t})$). In summary, while end-to-end speech recognition models use context, they can be enforced to make better use of it (i.e., through data augmentation or language model decoding).

10.1.2 *Direct or indirect dependencies?*

Again, the results in chapter 4 clearly show that speech recognition models depend on context. In fact, even words two or three positions from the target word help to reduce the word error rate. It might be tempting to conclude that the models have learned semantic features that span multiple words. However, this is not necessarily the case. Consider a simple example:

Assume that a speech recognition model fails to recognize the target word `her` with access to all the words but `she`. When `she` is included, the model is able to recognize the target word. One possible explanation is that the model has learned to match the grammatical gender of the pronoun `she` and the possessive adjective `her`. Another explanation is simply that the inclusion of `she` improved the estimate of `pulled`, which in turn improved the estimate of `her`. Thus, the question is whether context improves speech recognition models via direct or indirect dependencies. A simple graphical model illustrates this point in figure 23.

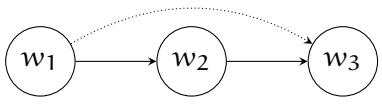


Figure 23: The results presented in chapter 4 show that context is important for speech recognition. However, it is not clear whether the improvements are a result of direct (dotted line) or indirect (solid lines) dependencies. Here, w_3 corresponds to the target word in the example above.

If the model only learns indirect dependencies, it would be analogous to say that the model only learns an implicit bigram language model. That is, if the terms in equation 85 was limited to $P(w_t|w_{t-1})$, although equation 85 is stated with reference to external language models. How to establish if a model learns direct or indirect dependencies warrants further research.

10.2 CHAPTER 5 REVISITED: LOOKING BEYOND TRANSCRIPTS

In chapter 5, a model for real-time question tracking and symptom detection in medical dialogues was proposed, analyzed, and discussed. With respect to the subsequent chapters, an important take-away is that text is more useful than speech for question tracking and symptom detection. However, when features inherently tied to speech (e.g., intonation) are relevant to the task, a multimodal approach can yield consistent improvements.

10.2.1 *Representation learning with speech recognition*

In chapter 9, speech representations learned without labeled data was shown to be competitive with speech recognition transcripts as input for spoken language understanding tasks. But even when plenty of labeled data is available, it may still not be the best solution to rely on transcripts as input for downstream tasks. As an alternative to the approach described in chapter 5, one could use the hidden representations of the speech recognizer rather than its output. Text as a representation is tailored to human perception. In the absence of exclamation and question marks, it does not capture features like intonation. Such markers are typically not available in speech recognition transcripts. Supervised representations from a speech recognizer, on the other hand, may encode both semantic and acoustic features beyond phonemic content.

Previous work has underlined the feasibility of learning representations with speech recognition. Palaskar et al. [212] show that acoustic word embeddings extracted from a speech recognition model yield good performance compared to text-based word embeddings on downstream tasks. Finally, Lugosch et al. [181] showed that pre-training an SLU model with speech recognition can improve performance. Thus, there is evidence that representations learned in speech recognition would offer an attractive alternative or extension to the multimodal approach described in chapter 5.

10.3 CHAPTER 7 REVISITED: DIGGING DEEPER IN WAV2VEC 2.0

In chapter 7, wav2vec 2.0 was compared to its predecessor and a bidirectional CPC model in the context of feature extraction for low-

resource speech recognition. The study emphasized the difficulties associated with training on wav2vec 2.0 representations compared to the other models. However, as will be elaborated below, extracting representations from the top layer of wav2vec 2.0 was a misguided choice.

10.3.1 *Choosing good representations from wav2vec 2.0*

Many representation models for speech use the output of a recurrent neural network or transformer encoder as input features for downstream tasks [52, 172, 176, 206, 241]. As outlined in the overview in chapter 6, most of these models try to reconstruct the input or another learned representation. The wav2vec 2.0 models target a learned quantized representation $\mathbf{q}_t$. The visualization of wav2vec 2.0 from chapter 7 is included in figure 24 for ease of reference. As such, these models resemble autoencoders. Indeed, Pasad et al. [215] show that the pre-trained wav2vec 2.0 models follow an autoencoder-like behavior. At the lower and upper layers of the transformer-based context network, the representations are similar to its input $\mathbf{z}_{1:T}$, measured with canonical correlation analysis (CCA). At the middle layers, the transformer representations deviate from the input. Intuitively, representations similar to input, or the quantized targets, are not ideal for downstream tasks.

Quantized representations can certainly encode relevant features (e.g., [45]). However, given the pre-training task described above, the primary objective for wav2vec 2.0 is not to encode useful features in $\mathbf{q}_t$. Instead, these features should be easily inferred from context. This might encourage sparsity in $\mathbf{q}_t$. In chapter 7, we saw that only few features are needed to explain most of the variance in $\mathbf{c}_t$ (figure 16), which supports this hypothesis to the extend that $\mathbf{c}_t$ and $\mathbf{q}_t$ are similar. Instead of extracting features from the last layer of the context network, the hidden layers of the model should be used. This is highlighted by application to spoken language understanding in chapter 9 and has also been shown in other recent work [10, 215].

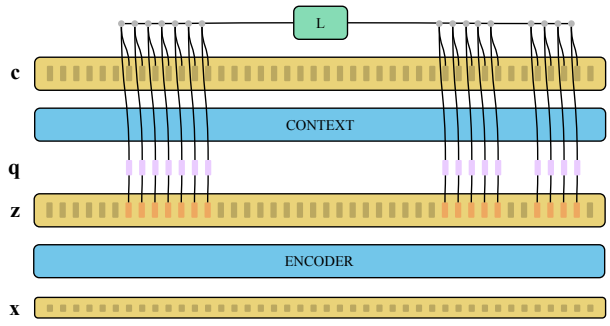


Figure 24: The wav2vec 2.0 framework [13] as illustrated in chapter 7.

10.3.2 *So why does PCA help?*

Again, the reasoning above suggests that $\mathbf{c}_t$ is a poor choice of representation. However, in chapter 7, we still saw that $\mathbf{c}_t$ yielded noticeable better results on the 10 hour subset than the other models when the feature space was decorrelated with PCA (table 12). One possible explanation is that the model does not learn to completely discard information encoded in the hidden layers of the context network. When $\mathbf{c}_t$ is decorrelated, the information pertaining to $\mathbf{q}_t$ is separated from the information that remains from the layers below.

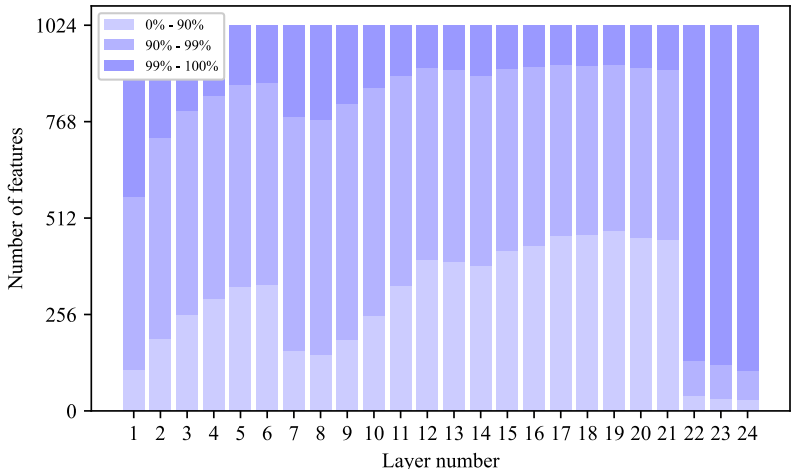


Figure 25: Explained variance ratio of each layer in the context network of wav2vec 2.0 (vox). Each bar highlights how many features are needed to explain 90%, 99% and 100% of the variance in a given layer after decorrelation with PCA. As in chapter 7, the PCA was computed on the 10 hour subset of LibriLight [136].

10.3.3 *Identifying good representations with PCA*

In chapter 7, the explained variance ratio of the feature space after decorrelating with PCA is used to analyze the learned representations. The analysis suggests that representations with few features explaining most of the variance can result in unstable training. Thus, the explained variance ratio may be used as a task agnostic indicator for extracting useful representations from wav2vec 2.0. Figure 25 shows a layer-wise plot of the explained variance ratio for the same levels (90%, 99% and 100%) as used in chapter 7. The bottom bars indicating 90% explained variance show a trend comparable to those found in other layer-wise explorations. Baevski et al. [10] found that the three top layers yielded poor results for phoneme classification while layers 10-21 gave the best results. Similarly, Pasad et al. [215] found that word meaning, measured as CCA similarity with text-based word embeddings, is greatest in layers 11-20.[2]

2 This was measured with a manual graph reading tool, so it might be slighlty off compared to the actual values from figure 1 in [215].

11

CONCLUSIONS AND OUTLOOK

In the introductory chapter, cardiac arrest detection in emergency calls was used as a motivational case. It formed the basis for discussing general challenges in conversational spoken language understanding (sec. 1.2) and defining the notion of semantic speech processing (sec. 1.4). In this concluding discussion, the studies presented in previous chapters will be reviewed in the context of this initial discussion, the general progress in the field, and future avenues of research. A complete description of the contributions made by each study was given in chapter 3 and will not be reiterated in detail here.

In **chapter 4**, end-to-end speech recognition models was analyzed in terms of their ability to utilize context. The findings resonate well with the development in the area of speech recognition. Three observations may be highlighted. First, the observation that models benefit from contextual information is a key insight behind successful data augmentation, such as SpecAugment [214], and masked pre-training, such as wav2vec 2.0 [13]. Second, the observation that attention-based models exhibit relative high context sensitivity is reflected in the growing popularity of self-attention architectures [266] for masking-based tasks [13, 50, 172, 175, 176]. Finally, connectionist temporal classification (CTC) represents a highly competitive end-to-end framework, in contrast to what has been found in previous studies [223]. Indeed, many recent works achieve state-of-the-art results using CTC for fine-tuning pre-trained models [13, 111], where it is particularly suitable as it requires minimal model adaptation.

Chapter 5 introduced a multi-modal approach for real-time question tracking and symptom detection. The study presents an interesting case for moving beyond speech recognition transcripts as input for downstream tasks. As described in later chapters, the progress in speech representation learning has seen more research focus on solving downstream tasks without speech recognition altogether. Certainly, it would be interesting to revisit the tasks studied in chapter 5 with this progress in mind.

It can be difficult to establish exactly where the advantage to a multimodal approach lies. For a task like question tracking, there is reason to believe that features beyond those that can be captured by text are useful, as also discussed in section 1.2.3. Intonation offers a plausible explanation for question tracking, but the speech signal may also offer a way to re-evaluate uncertainty relating to keywords in a

transcript. A speech recognition model may prefer to output words seen during training, as opposed to non-words, when faced with uncertainty. Thus, a model may alter the meaning of a sentence in a way that has little impact on the evaluation of transcript quality, but can be misleading for certain downstream tasks. How to explain predictions and understand learned features of speech processing models is an area that calls for more work. This is particularly important as semantic speech representation learning becomes ubiquitous.

Chapter 6 presented an overview of unsupervised neural speech representation learning. As emphasized in the overview, the wav2vec 2.0 framework [13], and masked pre-training in general, represent a breakthrough in low-resource speech recognition. The challenge associated with obtaining training data for conversational speech recognition, as discussed in section 1.2.1, has become much smaller. Even when no labeled in-domain data is available, this framework offers a viable solution [112].

The general idea for masked pre-training may seem simple; reconstruct masked parts of the input or another representation, given context. However, this methodology aligns very well with the general definition of semantics presented in section 1.4: *"semantic properties of a lexical item are fully reflected in appropriate aspects of the relations it contracts with actual and potential contexts"* ([63], p. 1). Thus, from a philosophical point-of-view, it may be difficult to see how the field should move on from here. Of course, context is more than just the neighboring words in a speech segment. The next wave of representation learning for speech incorporates multiple modalities, such as video [244, 245] or text [18]. Furthermore, how to construct targets for pre-training these models and how that affects the learned features is an avenue that warrants more research. This is of particular interest in the speech domain, where the input codes for speaker identity and emotional state, as well as the semantic content of the utterance.

Chapter 7 analyzed representations from the wav2vec and wav2vec 2.0 frameworks. As pointed out in section 10.3, the choice of representations is misguided in this analysis, but the topic of scaling representation models is still a relevant one. While increasing the number of parameters clearly tends to improve performance, it also comes with an increased demand for computational resources. The analysis in chapter 7 focuses on extracting representations from frozen models, as an alternative to fine-tuning them. In this scenario, representations can be stored on desk and used to train smaller downstream models, and thus, avoid a computational demanding forward-pass with each training step. However, when such models are deployed in real-world applications, the representation model is again needed. Academia is faced with similar problems. The focus on analyzing and

fine-tuning large pre-trained language models in text-based natural language processing will inevitably be mirrored in speech research. As this happens, it is important to address computational limitations. The extended analysis presented in section 10.3, highlights the significant amount of redundancy in models like wav2vec 2.0. Work on model distillation has recently shown great promise [39, 160] and making distilled speech models available to the research community is an important step towards more time and energy efficient research.

In **chapter 8**, stochastic and deterministic generative models were benchmarked in terms of model likelihood. In addition, the chapter presented a hierarchical latent variable model (LVM) for speech inspired by the Clockwork VAE [238]. In contrast to self-supervised models, LVMs have not yet seen the same success for representation learning. As such, they offer little guidance in terms of alleviating the need for labeled data, as discussed in section 1.2.1.

However, since LVMs learn a distribution over the training data, they are forced to encode all aspects of it. While this may be redundant for some tasks, others benefit from a broad variety of features. A hierarchical model that operates on multiple temporal scales, such as the Clockwork VAE, offers a natural way to encode different feature categories. Pronunciation might be learned at lower layers, speaker identity at the upper layers, and semantic features in between. Although some work has successfully separated speaker identity from content [114], models that can learn a deep hierarchy of features for speech remains an open challenge. The model presented in chapter 8 represents an effort to reignite this area of research.

In **chapter 9**, self-supervised speech representations were compared to speech recognition transcripts as input for multiple spoken language understanding tasks. The study includes a simplified version of the cardiac arrest detection task described in section 1.1. Across multiple trials, the study concluded that there is no significant performance difference between unsupervised representations and transcripts when used as input to the downstream model. This finding highlights a massive step towards overcoming the challenges described in the introduction of this thesis.

Results were mostly comparable for other classification tasks. However, for named entity recognition, the performance gap diminished as the amount of training data for the downstream task grew. With 960 hours of training data, speech recognition transcripts performed better than speech representations. For speech translation, the transcripts also presented the better option by a large margin. These findings suggest that semantic features are more accessible from text when plenty of data is available. This is true even when text comes in the form of error-ridden transcripts. It may be that discrete units

present regularities that are more easily learned by a simple downstream model. As highlighted in chapter 6, much recent work has leveraged quantization for speech representation learning [13, 111, 172]. For generative modeling in particular, quantization has proven vital for learning the structure of coherent speech [163, 265]. Whether quantized representations offer similar advantages for spoken language understanding is a direction that warrants further research.

Part V

APPENDIX

APPENDIX FOR CHAPTER 8

A.1 REPRODUCIBILITY STATEMENT

The source code used for the work presented in this paper will be made available before the conference. This code provides all details, practical and otherwise, needed to reproduce the results in this paper including data preprocessing, model training, model likelihood and latent space evaluation. The source code also includes scripts for downloading and preparing the LibriSpeech, LibriLight and TIMIT datasets. The LibriSpeech and LibriLight datasets are open source and can be downloaded with the preparation scripts. They are also available at https://www.openslr.org/12 and https://github.com/facebookresearch/libri-light, respectively. The TIMIT dataset is commercial and must be purchased and downloaded from https://catalog.ldc.upenn.edu/LDC93S1 before running the preparation script.

The stochastic latent variable models considered in this work do not provide an exact likelihood estimate nor an exact latent space representation. For the likelihood, they provide a stochastic lower bound and some variation in the reproduced likelihoods as well as latent representations must be expected between otherwise completely identical forward passes. This variance is fairly small in practice when averaging over large datasets such as those considered in this work. We seed our experiments to reduce the randomness to a minimum, but parts of the algorithms underlying the CUDA framework are stochastic for efficiency. To retain computational feasibility, we do not run experiments with a deterministic CUDA backend.

A.2 ETHICS STATEMENT

The work presented here fundamentally deals with automated perception of speech and generation of speech. These applications of machine learning potentially raise a number of ethical concerns. For instance, the these models might see possibly adverse use in automated surveillance and generation of deep fakes. To counter some of these effects, this work has focused on openness by using publicly available datasets for model development and benchmarking. Additionally, the work will open source the source code used to create these results. Ensuring the net positive effect of the development of these technologies is and must continue to be an ongoing effort.

We do not associate any significant ethical concerns with the datasets used in this work. However, one might note that the TIMIT dataset has somewhat skewed distributions in terms of gender and race diversity. Specifically, the male to female ratio is about two to one while the vast majority of speakers are Caucasian. Such statistics might have an effect of some ethical concern on downstream applications derived from such a dataset as also highlighted in recent research [152]. In LibriSpeech, there is an approximately equal number of female and male speakers while the diversity in race is unknown to the authors.

A.3 DATASETS

TIMIT TIMIT [87] is a speech dataset which contains 16 kHz recordings of 630 speakers of eight major dialects of American English, each reading ten phonetically rich sentences. It amounts to 6300 total recordings splits approximately in 3.94 hours of audio for training and 1.43 hours of audio for testing. No speakers or sentences in the test set are in the training set. The full train and test subsets of TIMIT are as in previous work [1, 56, 85]. We randomly sample 5% of the training set to use as a validation set. TIMIT includes temporally aligned annotations of phonemes and words as well as speaker metadata such as gender, height, age, race, education level and dialect region [87].

LIBRISPEECH AND LIBRILIGHT The LibriSpeech dataset [213] consists of readings of public domain audio books amounting to approximately 1000 h of audio. The data is derived from the LibriVox project. LibriLight [136] is a subset of LibriSpeech created as an automatic speech transcription (ASR) benchmark with limited or no supervision. We specifically train on the 100 h train-clean-100 subset of LibriSpeech and the 10 h subset of LibriLight. In all cases we evaluate on all the test splits dev-clean, dev-other, test-clean, test-other.

Both datasets represent the audio as 16 bit pulse code modulation (PCM) sampled at 16 000 Hz.

A.4 MODEL ARCHITECTURES

This section details model architectures. See appendix A.10 for graphical models and appendix A.5 for training details.

WAVENET We implement WaveNet as described in the original work [207] but use a discretized mixture of logistics as the output distribution as also done in other work [210]. Our WaveNet is not conditioned on any signal other than the raw waveform. The model applies the causal convolution directly to the raw waveform frames (i.e. one input channel). An alternative option that we did not examine is to replace

the initial convolution with an embedding lookup with a learnable vector for each waveform frame value.

LSTM The LSTM baseline uses an MLP encoder to embed the waveform subsegment $\boldsymbol{x}_{t:t+s-1}$ to a feature vector before feeding it to the LSTM cell. The encoder is similar to the parameterization of ϕ^{enc}_{vrnn} for the VRNN described above. The LSTM cell produces the hidden state $\boldsymbol{d}_t$ from $\boldsymbol{x}_{t:t+s-1}$ and passes it to a decoder. Like the encoder, the decoder is parameterized like ϕ^{dec}_{vrnn} of the VRNN. It outputs the waveform predictions $\boldsymbol{x}_{t+s:t+2s-1}$ from the hidden state $\boldsymbol{d}_t$. The LSTM model uses a single vanilla unidirectional LSTM cell.

VRNN We implement the VRNN as described in the original work [56] and verify that we can reproduce the original Gaussian likelihood TIMIT results. We replace the Gaussian output distribution with the DMoL.

SRNN We implement the VRNN as described in the original work [85] and verify that we can reproduce the original Gaussian likelihood TIMIT results. We replace the Gaussian output distribution with the DMoL.

CW-VAE We implement the CW-VAE based on the original work [238] but with some modifications also briefly described in section 8.2.6. We replace the encoder/decoder model architectures of the original work with architectures designed for waveform modeling. Specifically, the encoder and decoder are based on the Conv-TasNet [182] and uses similar residual block structure. However, contrary to the Conv-TasNet, we require downsampling factors larger than two. In order to achieve this we use strides of two in the separable convolution of each block. With e.g. six blocks we hence get an overall stride of $2^6 = 64$. We can then add additional blocks with unit stride. We also need to modify the residual connections that skip strided convolutions. Specifically, we replace the residual with a single convolution with stride equal to the stride used in the separable convolution. This convolution uses no nonlinearity and hence simply learns a local linear downsampling.

STCN We implement the STCN as described in the original work [1] and verify that we can reproduce the original Gaussian likelihood TIMIT results. We replace the Gaussian output distribution with the DMoL. We use the best-performing version of the STCN reported in the original paper, namedly the "STCN-dense" variant which conditions the observed variable on all five latent variables in the hierarchy. For the ablation experiment, we remove the bottom four latent variables. That is, we completely remove the corresponding four

small densely connected networks that parameterize the prior and posterior distributions based on deterministic representations of the WaveNet encoder. We keep the top most prior and posterior networks and use them to parameterize a latent variable of 256. This maintains the widest bottleneck of the model as well as almost all of the model's capacity.

ASR MODEL The ASR model used for the phoneme recognition experiments is a three-layered bidirectional LSTM. We apply temporal dropout between the LSTM layers and also after the final layer. Temporal dropout works similar to regular dropout but samples the entries of the hidden state to mask only once and apply it to all timesteps, i.e. masking $\boldsymbol{h}_t$ at vector index i for all t (and i). We mask by zeroing vector elements. We never mask the first timestep. We apply temporal dropout with masking probability of 0.3 for the 3.7h subset, 0.35 for the 1h subset and 0.4 for the 10m subset. The only difference in model architecture between the evaluation of different representations is the first affine transformation; from the dimensionality of the representation to the hidden state size of the LSTM. This gives rise to a very small difference in model capacity and parameter count which we find is negligible. We set the hidden unit size to 256.

A.5 TRAINING DETAILS

LIKELIHOOD BENCHMARK We implement all models and training scripts in PyTorch 1.9 [218]. For both datasets we use the Adam optimizer [146] with default parameters as given in PyTorch. We use learning rate $3e-4$ and no learning rate schedule. We use PyTorch automatic mixed precision (AMP) to significantly reduce memory consumption. We did not observe any significant difference in final model performance compared to full (32 bit) precision.

We train stateful models (LSTM, VRNN, SRNN and CW-VAE) on the full sequence lengths padding batches with zeros when examples are not of equal length. We sample batches such that they consist of examples that are approximately the same length to minimize the amount of computation wasted on padding.

For $s = 1$, we train stateless models (WaveNet, STCN) on random subsegments of the training examples and resample every epoch. This reduces memory requirements but does not bias the gradient. The subsequences are chosen to be of length 16000 which is larger than the receptive fields of the models and corresponds to one second of audio in TIMIT and LibriSpeech. For $s = 64$ and $s = 256$ we train the stateless models on the full example lengths similar to the stateful models since the receptive field is effectively s times larger and the shorter sequence length reduces memory requirements.

In testing, we evaluate on the full sequences. Due to memory constrains, for LibriSpeech, we need to split the test examples into subsegments since the average sequence length in Librispeech is about 4 times longer than that of TIMIT. Hence, we do multiple forward passes per test example, one for each of several subsegments. We carry along the internal state for models that are autoregressive in training (LSTM, VRNN, SRNN, CW-VAE) and define segments to overlap according to model architecture.

PHONEME RECOGNITION The ASR experiment consists of two stages: 1) pre-training of the unsupervised model and 2) training of the ASR model. The pre-training is done as for the likelihood benchmark above. The ASR model is trained using the Adam optimizer [146] with default parameters as given in PyTorch. We use learning rate $3e-4$ and no learning rate schedule.

For the spectrogram, WaveNet and the LSTM, we extract the representation only once and train the ASR model on these. Since the models are deterministic and do not parameterize distributions, this is the only option. For the LVMs, we resample the latent representation of a training example at every epoch. This is the most principled approach as these models parameterize probability distributions. Furthermore, using a single sample would be subject to artificially high variance in the representations while it is not straightforward to establish a sound mean representation for sequential models.

A.6 CONVERTING THE LIKELIHOOD TO UNITS OF BITS PER FRAME

Here we briefly describe how to compute a likelihood in units of bits per frame (bpf). In the main text, we use log to mean $\log_e$, but here we will be explicit. In general, conversion from nats to bits (i.e., from $\log_e$ to $\log_2$) is achieved by $\log_2(x) = \log_e(x)/\log_2(e)$. Remember that $\log_2 p(\boldsymbol{x}_{1:T})$ generally factorizes as $\sum_t \log_2 p(x_t|\cdot)$. In sequence modeling, it is important to remember that each example $\boldsymbol{x}^i$ must be weighted differently according the sequence length of that specific example. This is in contrast to computing bits per dimension in the image domain where images in a dataset are usually of the same dimensions. Thus, we compute the log-likelihood in bits per frame over the entire dataset as

$$\mathcal{L}(\boldsymbol{x}^i) = \frac{1}{\sum_i T_i} \sum_i \sum_t \log_2 p(\boldsymbol{x}_t^i) \ , \tag{86}$$

where i denotes the example index, T_i is the length of example $\boldsymbol{x}^i$ in waveform frames and t is the time index. If a single timestep $\boldsymbol{x}_t^i$ represents multiple waveform frames stacked with some stack size s, it is important to note that the sum over t only has T_i/s elements. For the LVMs, the term $\log_2 p(\boldsymbol{x}_t^i)$ is lower bounded by the ELBO in equation 70.

A.7 ADDITIONAL LIKELIHOOD RESULTS

TIMIT, μ-LAW, DMOL We provide additional results on TIMIT with audio represented as μ-law encoded PCM in table 21. Details are as presented in the main paper.

TIMIT, LINEAR, DMOL : We provide results on TIMIT with audio represented as linear PCM (raw PCM) in table 20. Except for the encoding, details are as for μ-law encoded TIMIT

TIMIT, LINEAR, GAUSSIAN We also provide some results on TIMIT with the audio instead represented as linear PCM (linearly encoded) and using Gaussian output distributions as has been done previously in the literature [1, 56, 85, 162]. We use $s = 200$ for comparability to the previous work. We provide the results in table 22 and include likelihoods reported in the literature for reference. For our models, we use the same architectures as before but replace the discretized mixture of logistics with either a Gaussian distribution or a mixture of Gaussian distributions.

We constrain the variance of the Gaussians used with our models to be at least $\sigma^2_{\text{min}} = 0.01^2$ in order to avoid the variance going to zero, the likelihood going to infinity and optimization becoming unstable.

From table 22 we note that the performance of the CW-VAE with Gaussian output distribution when modeling linear PCM (i.e. not μ-law encoded) does not compare as favorably to the other baselines as it did with the discretized mixture of logistics distribution. We hypothesize that this has to do with using a Gaussian output distribution in latent variable models which, as has been reported elsewhere [188], leads to a likelihood function that is unbounded above and can grow arbitrarily high. We discuss this phenomenon in further detail in section A.8.

We specifically hypothesize that models that are autoregressive in the observed variable (VRNN, SRNN, Stochastic WaveNet, STCN) are well-equipped to utilize local smoothness to put very high density on the correct next value and that this in turn leads to a high degree of exploitation of the unboundedness of the likelihood. Not being autoregressive in the observed variable, the CW-VAE cannot exploit this local smoothness in the same way. Instead, the reconstruction is conditioned on a stochastic latent variable, $p(\boldsymbol{x}_t|\boldsymbol{z}_t^1)$, which introduces uncertainty and likely larger reconstruction variances.

s	**Model**	**Configuration**	$\mathcal{L}$ **[bpf]**
1	Uniform	Uninformed	16.00
1	DMoL	Optimal	10.70
-	FLAC	Linear PCM	8.582
1	Wavenet	$D_C = 96$	**7.246**
1	LSTM	$D_d = 256, L = 1$	7.295
1	VRNN	$D_z = 256$	$\leqslant$7.316
1	SRNN	$D_z = 256$	$\leqslant$7.501
1	STCN		$\leqslant$9.970
64	WaveNet	$D_c = 96$	8.402
64	LSTM	$D_d = 256, L = 1$	8.357
64	VRNN	$D_z = 256$	$\leqslant$8.103
64	SRNN	$D_z = 256$	$\leqslant$8.036
64	CW-VAE	$D_z = 96, L = 1$	$\leqslant$7.989
64	STCN		$\leqslant$**7.768**
256	WaveNet	$D_c = 96$	9.018
256	LSTM	$D_d = 256, L = 1$	8.959
256	VRNN	$D_z = 256$	$\leqslant$8.739
256	SRNN	$D_z = 256$	$\leqslant$8.674
256	CW-VAE	$D_z = 96, L = 1$	$\leqslant$8.406
256	STCN		$\leqslant$**8.196**

Table 20: Model likelihoods on TIMIT represented as a **16bit linear PCM**, obtained by different latent variable models and compared to autoregressive baselines all using a discretized mixture of logistics with 10 components as output distribution. Likelihoods are given in units of bits per frame (bpf) and obtained by normalizing the total likelihood of each sequence with the individual sequence length and then averaging over the dataset. The STCN converges to a poor local minimum and sometimes diverges when modeling linear PCM with $s = 1$.

s	Model	Configuration	$\mathcal{L}$ [bpf]
1	Wavenet	$D_C = 16$	11.27
1	Wavenet	$D_C = 24$	11.14
1	Wavenet	$D_C = 32$	11.03
1	Wavenet	$D_C = 96$	10.88
1	Wavenet	$D_C = 128$	10.98
1	Wavenet	$D_C = 160$	10.91
1	LSTM	$D_d = 128, L = 1$	11.40
1	LSTM	$D_d = 256, L = 1$	11.11
1	VRNN	$D_z = 256$	$\leqslant$11.09
1	SRNN	$D_z = 256$	$\leqslant$11.19
4	LSTM	$D_d = 256, L = 1$	11.65
16	LSTM	$D_d = 256, L = 1$	12.54
16	LSTM	$D_d = 256, L = 2$	12.54
16	LSTM	$D_d = 256, L = 3$	12.44
64	WaveNet	$D_c = 96$	13.30
64	LSTM	$D_d = 96, L = 1$	13.49
64	LSTM	$D_d = 96, L = 2$	13.46
64	LSTM	$D_d = 96, L = 3$	13.40
64	LSTM	$D_d = 256, L = 1$	13.27
64	LSTM	$D_d = 256, L = 2$	13.29
64	LSTM	$D_d = 256, L = 3$	13.31
64	LSTM	$D_d = 512, L = 1$	13.37
64	LSTM	$D_d = 512, L = 2$	13.37
64	LSTM	$D_d = 512, L = 3$	13.41
64	VRNN	$D_z = 96$	$\leqslant$12.93
64	VRNN	$D_z = 256$	$\leqslant$12.54
64	SRNN	$D_z = 96$	$\leqslant$12.87
64	SRNN	$D_z = 256$	$\leqslant$12.42
64	CW-VAE	$D_z = 96, L = 1$	$\leqslant$12.44
64	CW-VAE	$D_z = 96, L = 2$	$\leqslant$12.17
64	CW-VAE	$D_z = 96, L = 3$	$\leqslant$12.15
64	CW-VAE	$D_z = 256, L = 2$	$\leqslant$12.10
256	WaveNet	$D_c = 96$	14.11
256	LSTM	$D_d = 256, L = 1$	14.20
256	LSTM	$D_d = 256, L = 2$	14.17
256	LSTM	$D_d = 256, L = 3$	14.26
256	VRNN	$D_z = 96$	$\leqslant$13.51
256	VRNN	$D_z = 256$	$\leqslant$13.27
256	SRNN	$D_z = 96$	$\leqslant$13.28
256	SRNN	$D_z = 256$	$\leqslant$13.14
256	CW-VAE	$D_z = 96, L = 1$	$\leqslant$13.11
256	CW-VAE	$D_z = 96, L = 2$	$\leqslant$12.97
256	CW-VAE	$D_z = 96, L = 3$	$\leqslant$12.87

Table 21: Model likelihoods on TIMIT represented as a **16bit μ-law encoded PCM**, obtained by different latent variable models and compared to autoregressive baselines all using a discretized mixture of logistics with 10 components as output distribution. Likelihoods are given in units of bits per frame (bpf) and obtained by normalizing the total likelihood of each sequence with the individual sequence length and then averaging over the dataset.

s	**Model**	**Configuration**	$\mathcal{L}$ **[nats]**
1	WaveNet	Normal	119656
1	WaveNet	GMM-2	120699
1	WaveNet	GMM-20	121681
200	WaveNet [1]	GMM-20	30188
200	WaveNet [1]	Normal	-7443
200	Stochastic WaveNet* [162]	Normal	$\geqslant$72463
200	VRNN [56]	Normal	$\approx$28982
200	SRNN [85]	Normal	$\geqslant$60550
200	STCN [1]	GMM-20	$\geqslant$69195
200	STCN [1]	Normal	$\geqslant$64913
200	STCN-dense [1]	GMM-20	$\geqslant$71386
200	STCN-dense [1]	Normal	$\geqslant$70294
200	STCN-dense-large [1]	GMM-20	$\geqslant$77438
200	CW-VAE*	$L = 1, D_z = 96$, Normal	$\geqslant$41629

Table 22: Model likelihoods on TIMIT represented as **globally normalized 16bit linear PCM**. Likelihoods are given in units of nats and obtained by summing the likelihood over time and over all examples in the dataset and dividing by the total number of examples. In the table, Normal refers to using a Gaussian likelihood and GMM refers to using a Gaussian Mixture Model likelihood with 20 components. Models with asterisks * are our implementations while remaining results are as reported in the referenced work.

A.8 ADDITIONAL DISCUSSION ON GAUSSIAN LIKELIHOODS IN LVMS

As noted in section A.7, we constrain the variance of the output distribution of our models to be $\sigma^2_{\text{min}} = 0.01^2$ for the additional results on TIMIT with Gaussian outputs. This limits the maximum value attainable by the prediction/reconstruction density of a single waveform frame x_t.

Specifically, we can see that since

$$\log p(x_t|\cdot) = \log \mathcal{N}\left(x_t; \mu_t, \max\left\{\sigma^2_{\text{min}}, \sigma^2_t\right\}\right) \ , \tag{87}$$

the best prediction/reconstruction density is achieved when $\sigma^2 \leqslant \sigma^2_{\text{min}}$ and $\mu = x_t$. Here $\cdot$ indicates any variables we might condition on such as the previous input frame x_{t-1} or some latent variables. We can evaluate this best case scenario for $\sigma^2_{\text{min}} = 0.01^2$,

$$\begin{aligned}
\log \mathcal{N}\left(x_t; x_t, \sigma^2_{\text{min}}\right) &= -\frac{1}{2}\log 2\pi - \frac{1}{2}\log \sigma^2_{\text{min}} - \frac{1}{2\sigma^2_{\text{min}}}(x_t - x_t) \\
&= -\frac{1}{2}\log 2\pi - \frac{1}{2}\log 0.01^2 \\
&= 3.686 \ .
\end{aligned} \tag{88}$$

Hence, with perfect prediction/reconstruction and the minimal variance (0.01^2), a waveform frame contributes to the likelihood with 3.686 nats. With an average test set example length of 49 367.3 frames frames this leads to a best-case likelihood of 181967. We provide a list of maximally attainable Gaussian likelihoods on TIMIT for different minimal variances in table 23. One can note that the maximal likelihood at $\sigma^2_{\text{min}} = 0.1^2$ is lower than the likelihoods achieved by some models in table 22. This indicates that the models learn to use very small variances in order to increase the likelihood.

σ^2_{min}	max $\mathcal{L}$
1^2	-45367
0.5^2	-11146
0.1^2	68307
0.05^2	102525
0.01^2	181979
0.005^2	216198
0.001^2	295651

Table 23: The highest possible Gaussian log-likelihoods (max $\mathcal{L}$) attainable on the TIMIT test set as computed by equation 87 with different values of the minimum variance σ^2_{min}.

A.9 ADDITIONAL DISCUSSION ON THE CHOICE OF OUTPUT DISTRIBUTION

The DMoL uses a discretization of the continuous logistic distribution to define a mixture model over a discrete random variable. This allows it to parameterize multimodal distributions which can express ambiguity about the value of $\boldsymbol{x}_t$. The model can learn to maximize likelihood by assigning a bit of probability mass to multiple potential values of $\boldsymbol{x}_t$.

While this is well-suited for autoregressive modeling, for which the distribution was developed, the potential multimodality poses a challenge for non-autoregressive latent variable models which independently sample multiple neighboring observations at the output. In fact, if multiple neighboring outputs defined by the subsequence $\boldsymbol{x}_{t_1:t_2}$ have multimodal $p(\boldsymbol{x}_t|\cdot)$, we risk sampling a subsequence where each neighboring value expresses different potential realities, independently.

Interestingly, most work on latent variable models with non-autoregressive output distributions seem to ignore this fact and simply employ the mixture distribution with 10 mixture components [43, 185, 264]. However, given the empirically good results of latent variable models for image generation, this seems to have posed only a minor problem in practice. We speculate that this is due to the high degree of similarity between neighbouring pixels in images. I.e. if the neighboring pixels are nuances of red, then, in all likelihood, so is the central pixel.

In the audio domain, however, neighbouring waveform frames can take wildly different values, especially at low sample rates. Furthermore, waveforms exhibit a natural symmetry between positive and negative amplitudes. Hence, it seems plausible that multimodality may pose a larger problem in non-autoregressive speech generation by causing locally incoherent samples than it seems to do in image modelling.

A.10 ADDITIONAL GRAPHICAL MODELS

In figure 26 we show the graphical model of the recurrent cell of the CW-VAE for a single time step. As noted in [238], this cell is very similar to the one of the Recurrent State Space Model (RSSM) [96].

In figure 27 we show the unrolled graphical models of a three-layered CW-VAE with $k_1 = 1$ and $c = 2$ yielding $k_2 = 2$ and $k_3 = 4$. We show both the generative and inference models and highlight in blue the parameter sharing between the two models due to top-down inference.

In figure 28 we show the graphical models of the STCN [1] at a single timestep. The model has three layers and shares the parameters of the WaveNet encoder between the inference and generative models.

In figure 29 we illustrate the unrolled graphical models of the inference and generative models of the VRNN [56]. We include the deterministic variable $\boldsymbol{d}_t$ in order to illustrate the difference to other latent variable models.

Likewise, in figure 30 we illustrate the unrolled graphical models the SRNN [85].

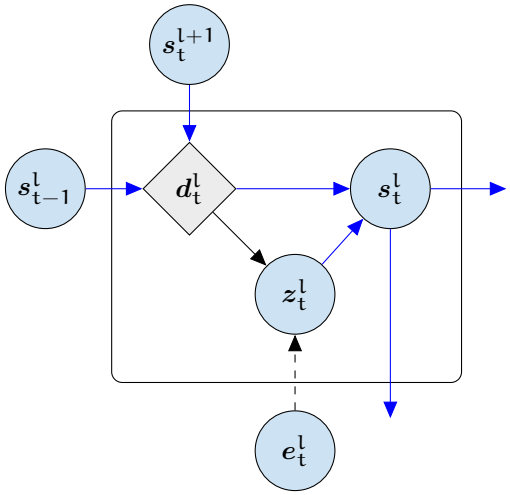
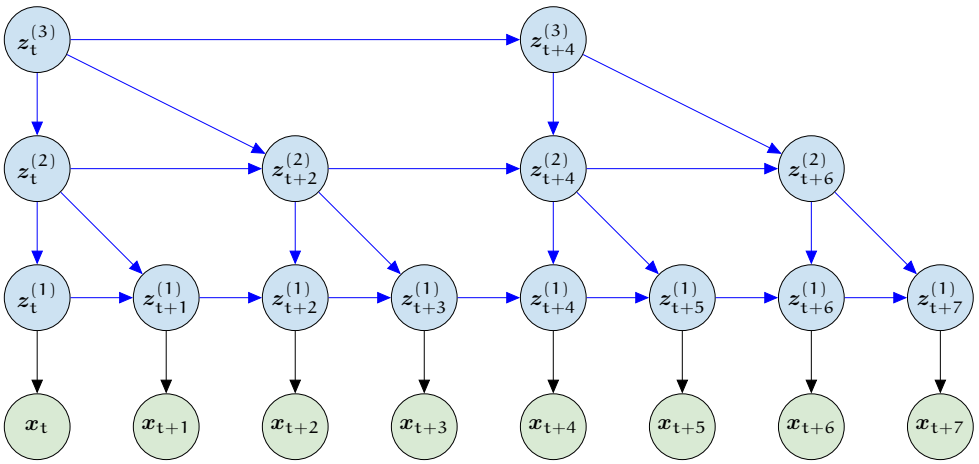


Figure 26: CW-VAE cell state $\boldsymbol{s}_t^l$ update. The cell state is given as $\boldsymbol{s}_t^l = (\boldsymbol{z}_t^l, \boldsymbol{d}_t^l)$ where $\boldsymbol{d}_t^l$ is the deterministic hidden state of a Gated Recurrent Unit [44]. The vector $\boldsymbol{e}_t^l$ is computed from $\boldsymbol{x}_t$ by the encoder network which outputs L encodings, one for each latent variable, similar to that of a Ladder VAE [250]. All blue arrows are shared between generation and inference. The dashed arrow is used only during inference. The solid arrow has unique transformations during inference and generation.

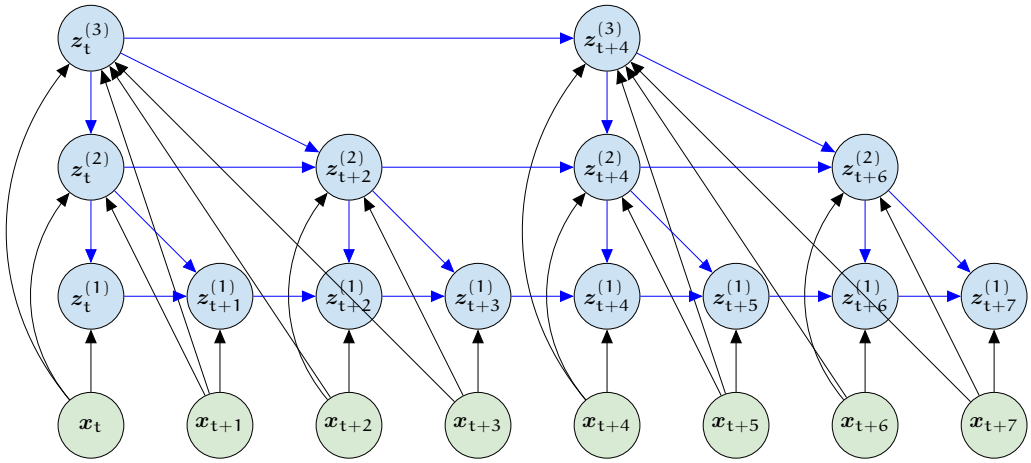
A.11 ADDITIONAL LATENT EVALUATION

We visualize the performance of a k-nearest-neighbour classifier for classification of speaker gender and height in figure 31. The classifier is fitted to time-averaged latent representations and Mel-features. We divide the height into three classes: below 175 cm, above 185 cm and in-between. Compared to phonemes, the gender and height of a speaker are global attributes that affect the entire signal. In both cases, we see improved performance from using the learned latent space over Mel-features. Notably, $\boldsymbol{z}^2$ is outperformed by the Mel-features for gender identification which may indicate that $\boldsymbol{z}^2$ learns to ignore this attribute compared to $\boldsymbol{z}^1$.

We provide some additional latent space clustering of speaker gender in figure 32 and of speaker height in figure 33.



(a)



(b)

Figure 27: CW-VAE [238] generative model $p(\boldsymbol{x}, \boldsymbol{z})$ in (a) and inference model $q(\boldsymbol{z}|\boldsymbol{x})$ in (b) for a three-layered model with $k_1 = 1$ and $c = 2$ giving $k_2 = 2$ and $k_3 = 4$ unrolled over eight steps in the observed variable. Blue arrows are (mostly) shared between the inference and generative models. See figure 26 for a detailed graphical model expanding on the latent nodes z_t^1 and parameter sharing.

All results presented here are obtained with a 2-layered CW-VAE trained on μ-law encoded PCM similar to the one in table 13.

A.12 DISTRIBUTION OF PHONEME DURATION IN TIMIT

In figure 34 we plot a boxplots of the duration of each phoneme in the TIMIT dataset. We do this globally as well as for a single speaker to show that phoneme duration can vary between individual speakers.

A description of the phonemes used for the TIMIT dataset can be found at https://catalog.ldc.upenn.edu/docs/LDC93S1/PHONCODE.TXT.

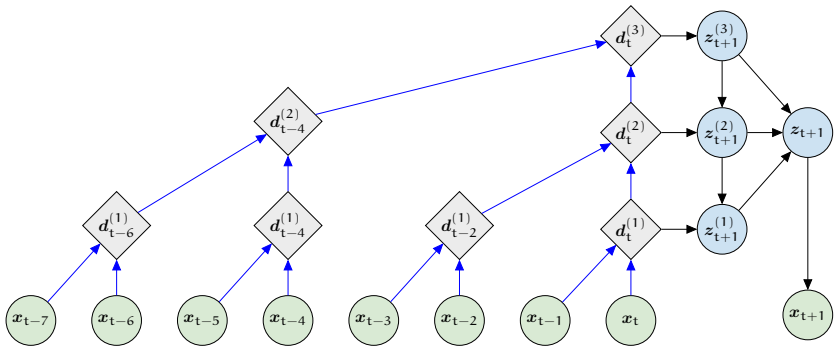
A.13 MODEL SAMPLES AND RECONSTRUCTIONS

We provide samples and reconstructions for some of the models considered here at the following URL: https://doi.org/10.5281/zenodo.5911899.

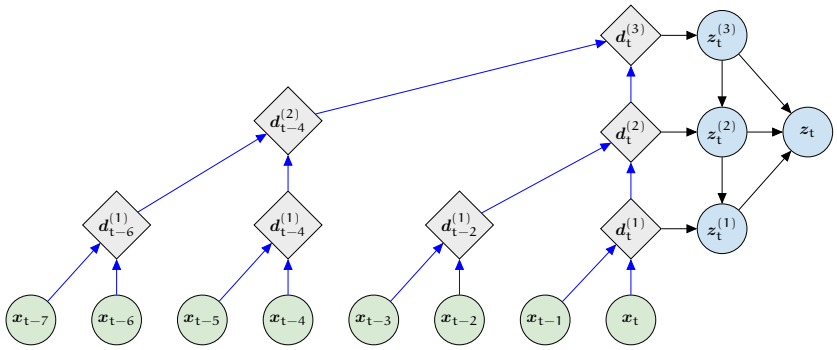
The samples are generated from the prior of Clockwork VAE, SRNN and VRNN and from a WaveNet by conditioning on pure zeros. All models are configured as those reported in table 13.

Importantly, the samples are unconditional. Hence they are *not* reconstructions inferred from a given input nor are they conditioned on any auxiliary data like text.

Although sample quality is a somewhat subjective matter, we find the quality of the unconditional Clockwork VAE to be better than those of our VRNN and SRNN. WaveNet is known to produce samples with intelligible speech when conditioned on e.g. text, but unconditional samples from WaveNet lack semantic content such as words similar to VRNN, SRNN and Clockwork VAE.



(a)



(b)

Figure 28: STCN [1] generative model $p(\boldsymbol{x}, \boldsymbol{z})$ in (a) and inference model $q(\boldsymbol{z}|\boldsymbol{x})$ in (b) for a single time-step. The WaveNet autoregressive encoder is shared between generative and inference models. It is depicted here with only one stack of three layers in order to illustrate the dilated convolution with limited space. In practice, the model uses ten layers in each of five stacks/cycles resulting in a much larger receptive field. Importantly, the model parameterizes the five latent variables using the last deterministic representation $\boldsymbol{d}^{(l)}$ from each stack, i.e. only every fifth l starting from $l = 5$ and ending at $l = 25$. Note that the generative model uses the prior to transform the WaveNet hidden states $\boldsymbol{d}_t^{(l)}$ into the latent variable $\boldsymbol{z}_{t+1}^{(l)}$ one step ahead in time compared to the approximate posterior which infers $\boldsymbol{z}_t^{(l)}$. Also note that $\boldsymbol{z}_t$ is constructed by concatenating all $\boldsymbol{z}_t^{(l)}$. The original paper explores setting $\boldsymbol{z}_t$ equal to $\boldsymbol{z}_t^{(1)}$. The best-performing STCN for speech, which also the one we implement, uses a WaveNet decoder to predict $\boldsymbol{x}_{t+1}$ from a sequence of $\boldsymbol{z}_t$ rather than a per-timestep transform. Blue arrows are shared between the inference and generative models.

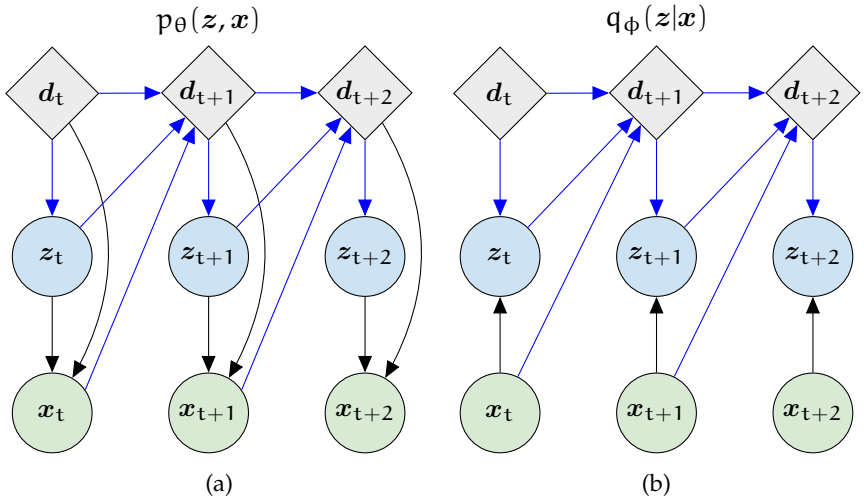


Figure 29: VRNN [56] generative model $p(x, z)$ in (a) and inference model $q(z|x)$ in (b) unrolled over three steps in the observed variable. Blue arrows are shared between the inference and generative models.

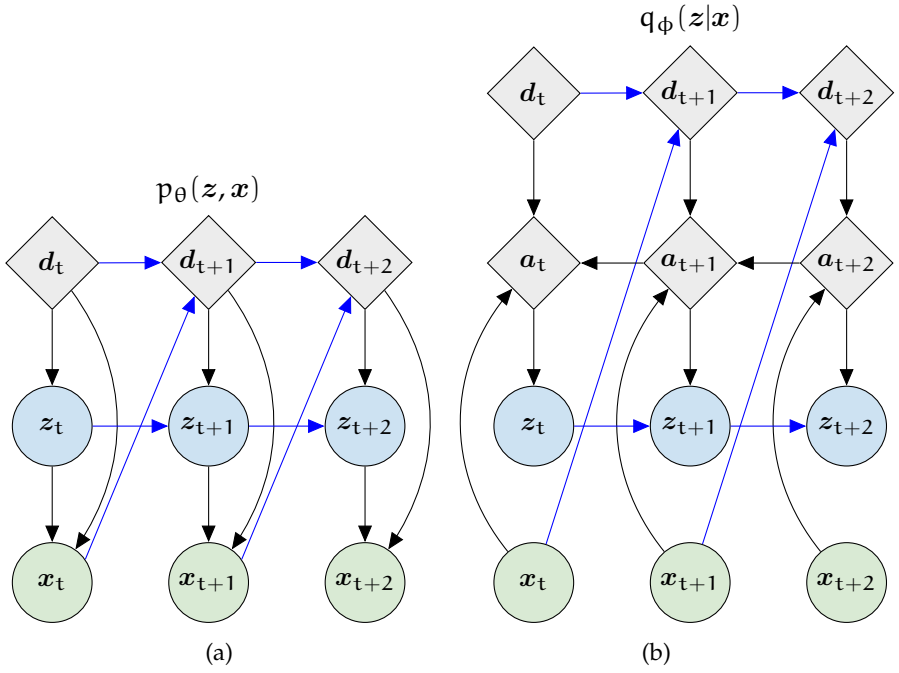


Figure 30: SRNN [85] generative model $p(x, z)$ in (a) and inference model $q(z|x)$ in (b) unrolled over three steps in the observed variable. Blue arrows are shared between the inference and generative models.

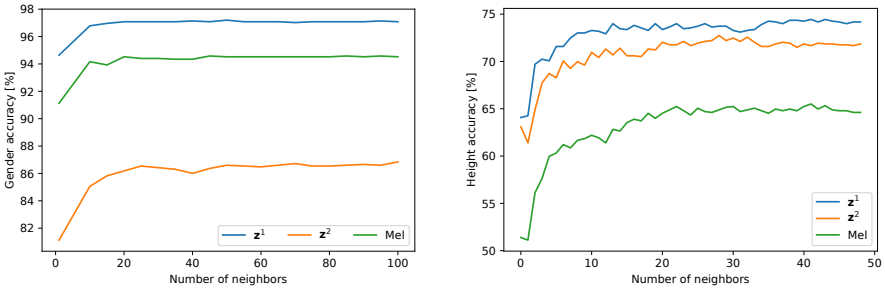
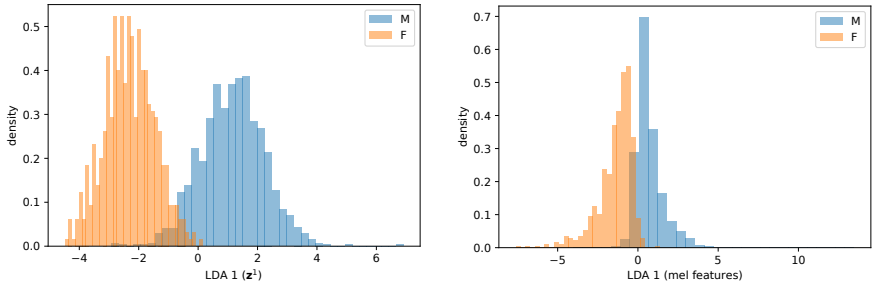
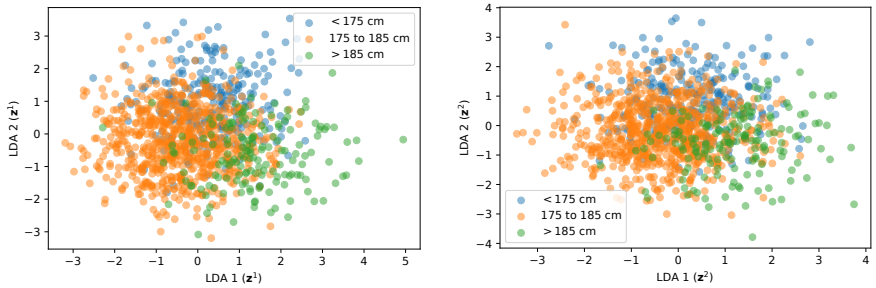


Figure 31: Leave-one-out k-nearest-neighbor accuracy with different k for (left) the speaker's gender and (right) the height of male speakers (female speakers yield a similar result).



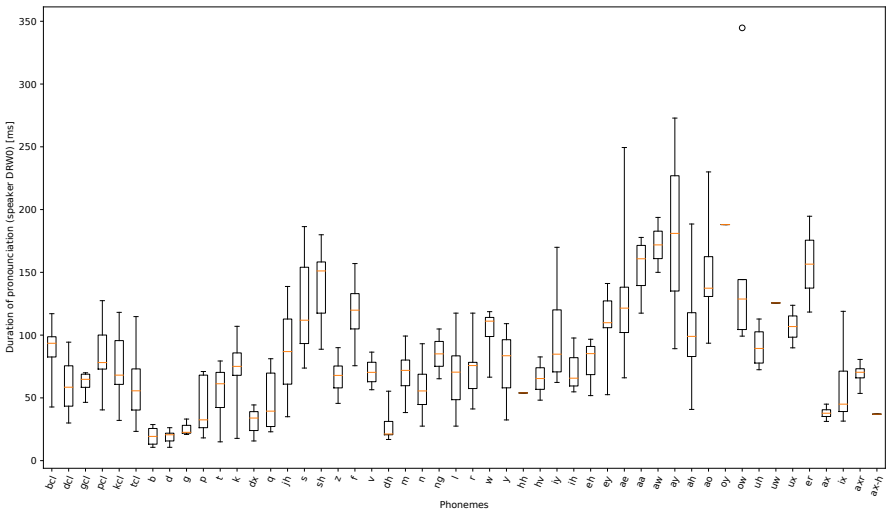
(a) (b)

Figure 32: Clustering of speaker gender in an one-dimensional linear subspace defined by a linear discriminant analysis of the CW-VAE latent space and of a time-averaged mel spectrogram. The total overlap is slightly smaller in the subspace of the CW-VAE latent space and the separation between the distribution peaks is larger.

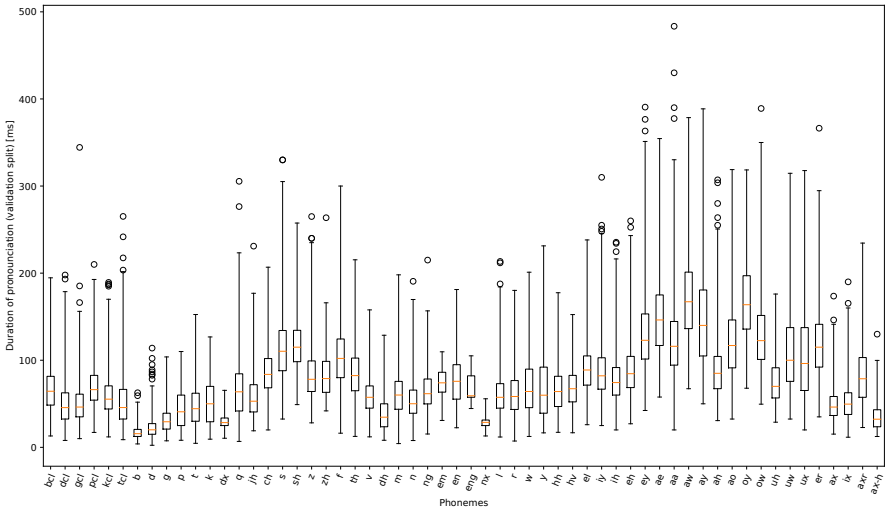


(a) (b)

Figure 33: Clustering of speaker height in an two-dimensional linear subspace defined by a linear discriminant analysis of the CW-VAE latent space.



(a)



(b)

Figure 34: Boxplots of the duration of the pronunciation of phonemes in TIMIT for a specific speaker DRW0 in (a) and globally in (b). Not all phonemes are pronounced by speaker DRW0 over the course of their 10 test set sentences and hence they are missing from the x-axis compared to the global durations.

BIBLIOGRAPHY

[1] Emre Aksan and Otmar Hilliges. "STCN: Stochastic Temporal Convolutional Networks." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2019.

[2] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. "Deep speech 2: End-to-end speech recognition in english and mandarin." In: *International conference on machine learning*. PMLR. 2016, pp. 173–182.

[3] Douglas Coimbra de Andrade, Sabato Leo, Martin Loesener Da Silva Viana, and Christoph Bernkopf. "A Neural Attention Model for Speech Command Recognition." In: *arXiv preprint arXiv:1808.08929* (2018).

[4] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. "Common Voice: A Massively-Multilingual Speech Corpus." In: *Proceedings of the Conference on Language Resources and Evaluation (LREC)*. 2020, pp. 4211–4215.

[5] Siddhant Arora, Alissa Ostapenko, Vijay Viswanathan, Siddharth Dalmia, Florian Metze, Shinji Watanabe, and Alan W Black. "Rethinking End-to-End Evaluation of Decomposable Tasks: A Case Study on Spoken Language Understanding." In: *arXiv preprint arXiv:2106.15065* (2021).

[6] Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "Explaining recurrent neural network predictions in sentiment analysis." In: *EMNLP Workshop: Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis* (2017).

[7] Leila Arras, Ahmed Osman, Klaus-Robert Müller, and Wojciech Samek. "Evaluating Recurrent Neural Network Explanations." In: *ACL Workshop: BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (2019).

[8] Leonardo Badino, Claudia Canevari, Luciano Fadiga, and Giorgio Metta. "An Auto-Encoder Based Approach to Unsupervised Learning of Subword Units." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 7634–7638.

[9] Alexei Baevski, Michael Auli, and Abdelrahman Mohamed. "Effectiveness of self-supervised pre-training for speech recognition." In: *arXiv preprint arXiv:1911.03912* (2019).

[10] Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. "Unsupervised speech recognition." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34. 2021.

[11] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. "data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language." In: *arXiv preprint arXiv:2202.03555* (2022).

[12] Alexei Baevski, Steffen Schneider, and Michael Auli. "vq-wav2vec: Self-Supervised Learning of Discrete Speech Representations." In: *Proceedings of the International Conference on Learning Representations (ICLR)* (2019).

[13] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020.

[14] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." In: *International Conference on Learning Representations (ICLR)*. 2015.

[15] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. "End-to-end attention-based large vocabulary speech recognition." In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 4945–4949.

[16] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. "Multimodal machine learning: A survey and taxonomy." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.2 (2018), pp. 423–443.

[17] Sameer Bansal, Herman Kamper, Adam Lopez, and Sharon Goldwater. "Towards speech-to-text translation without speech recognition." In: *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. 2017, pp. 474–479.

[18] Ankur Bapna, Yu-an Chung, Nan Wu, Anmol Gulati, Ye Jia, Jonathan H Clark, Melvin Johnson, Jason Riesa, Alexis Conneau, and Yu Zhang. "SLAM: A Unified Encoder for Speech and Language Modeling via Speech-Text Joint Pre-Training." In: *arXiv preprint arXiv:2110.10329* (2021).

[19] David Bartholomew, Martin Knott, and Irini Moustaki. "Latent Variable Models and Factor Analysis: A Unified Approach." In: Wiley Series in Probability and Statistics (1987).

[20] Eric Battenberg, Jitong Chen, Rewon Child, Adam Coates, Yashesh Gaur Yi Li, Hairong Liu, Sanjeev Satheesh, Anuroop Sriram, and Zhenyao Zhu. "Exploring neural transducers for end-to-end speech recognition." In: *Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE. 2017, pp. 206–213.

[21] Justin Bayer, Maximilian Soelch, Atanas Mirchev, Baris Kayalibay, and Patrick van der Smagt. "Mind the Gap when Conditioning Amortised Inference in Sequential Latent-Variable Models." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2020.

[22] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8 (2013), pp. 1798–1828.

[23] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. "Estimating or propagating gradients through stochastic neurons for conditional computation." In: *arXiv preprint arXiv:1308.3432* (2013).

[24] Homanga Bharadhwaj. "Layer-wise relevance propagation for explainable deep learning based speech recognition." In: *International Symposium on Signal Processing and Information Technology (ISSPIT)*. IEEE. 2018, pp. 168–174.

[25] Saurabhchand Bhati, Jesús Villalba, Piotr Żelasko, Laureano Moro-Velazquez, and Najim Dehak. "Segmental Contrastive Predictive Coding for Unsupervised Word Segmentation." In: *arXiv preprint arXiv:2106.02170* (2021).

[26] Saurabhchand Bhati, Jesús Villalba, Piotr Żelasko, Laureano Moro-Velazquez, and Najim Dehak. "Unsupervised Speech Segmentation and Variable Rate Representation Learning using Segmental Contrastive Predictive Coding." In: *arXiv preprint arXiv:2110.02345* (2021).

[27] Joachim Bingel and Anders Søgaard. "Identifying beneficial task relations for multi-task learning in deep neural networks." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2017, pp. 164–169.

[28] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Publication Title: New York: springer. Springer, 2006. ISBN: 978-0-387-31073-2.

[29] Stig Nikolaj Blomberg, Fredrik Folke, Annette Kjær Ersbøll, Helle Collatz Christensen, Christian Torp-Pedersen, Michael R Sayre, Catherine R Counts, and Freddy K Lippert. "Machine learning as a supportive tool to recognize cardiac arrest in emergency calls." In: *Resuscitation* 138 (2019), pp. 322–329.

[30] Lasse Borgholt, Jakob D Havtorn, Anders Søgaard Željko Agic, Lars Maaløe, and Christian Igel. "Do End-to-End Speech Recognition Models Care About Context?" In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020, pp. 4352–4356.

[31] Lasse Borgholt, Jakob Drachmann Havtorn, Mostafa Abdou, Joakim Edin, Lars Maaløe, Anders Søgaard, and Christian Igel. "Do We Still Need Automatic Speech Recognition for Spoken Language Understanding?" In: *arXiv preprint arXiv:2111.14842* (2021).

[32] Lasse Borgholt, Tycho MS Tax, Jakob D Havtorn, Lars Maaløe, and Christian Igel. "On Scaling Contrastive Representations for Low-Resource Speech Recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 3885–3889.

[33] Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. "Generating Sentences from a Continuous Space." In: *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. 2016, pp. 10–21.

[34] Karlheinz Brandenburg, Ernst Eberlein, Heinz Gerhäuser, Bernhard Grill, Jürgen Herre, and Harald Popp. *MPEG-2 Audio Layer III (MP3)*. Germany, 1998.

[35] D Burton, J Shore, and J Buck. "A generalization of isolated word recognition using vector quantization." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vol. 8. IEEE. 1983, pp. 1021–1024.

[36] Michael A. Carlin, Samuel Thomas, Aren Jansen, and Hynek Hermansky. "Rapid Evaluation of Speech Representations for Spoken Term Discovery." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2011, pp. 821–824.

[37] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 4960–4964.

[38] William Chan, Yu Zhang, Quoc Le, and Navdeep Jaitly. "Latent Sequence Decompositions." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2017.

[39] Heng-Jui Chang, Shu-wen Yang, and Hung-yi Lee. "DistilHuBERT: Speech Representation Learning by Layer-wise Distillation of Hidden-unit BERT." In: *arXiv preprint arXiv:2110.01900* (2021).

[40] Kumar Chellapilla, Sidd Puri, and Patrice Simard. "High performance convolutional neural networks for document processing." In: *Tenth International Workshop on Frontiers in Handwriting Recognition*. 2006.

[41] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing." In: *arXiv preprint arXiv:2110.13900* (2021).

[42] Po-Han Chi, Pei-Hung Chung, Tsung-Han Wu, Chun-Cheng Hsieh, Yen-Hao Chen, Shang-Wen Li, and Hung-yi Lee. "Audio albert: A Lite BERT for Self-Supervised Learning of Audio Representation." In: *IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2021, pp. 344–350.

[43] Rewon Child. "Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2021.

[44] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. arXiv preprint arXiv:1409.1259. 2014.

[45] Jan Chorowski, Nanxin Chen, Ricard Marxer, Hans Dolfing, Adrian Łańcucki, Guillaume Sanchez, Tanel Alumäe, and Antoine Laurent. "Unsupervised Neural Segmentation and Clustering for Unit Discovery in Sequential Data." In: *NeurIPS Workshop: Perception as Generative Reasoning* (2019).

[46] Jan Chorowski and Navdeep Jaitly. "Towards better decoding and language model integration in sequence to sequence models." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2017, pp. 523–527.

[47] Jan Chorowski, Ron J. Weiss, Samy Bengio, and Aäron van den Oord. "Unsupervised speech representation learning using WaveNet autoencoders." In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2019).

[48] Yu-An Chung and James Glass. "Learning Word Embeddings from Speech." In: *NeurIPS ML4Audio Workshop*. 2017.

[49] Yu-An Chung and James Glass. "Speech2Vec: A Sequence-to-Sequence Framework for Learning Word Embeddings from Speech." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2018, pp. 811–815.

[50] Yu-An Chung and James Glass. "Generative Pre-training for Speech With Autoregressive Predictive Coding." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 3497–3501.

[51] Yu-An Chung and James Glass. "Improved Speech Representations with Multi-Target Autoregressive Predictive Coding." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2020, pp. 146–150.

[52] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. "An Unsupervised Autoregressive Model for Speech Representation Learning." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 146–150.

[53] Yu-An Chung, Hao Tang, and James Glass. "Vector-Quantized Autoregressive Predictive Coding." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020, pp. 3760–3764.

[54] Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. "Audio Word2Vec: Unsupervised Learning of Audio Segment Representations Using Sequence-to-Sequence Autoencoder." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2016, pp. 765–769.

[55] Yu-An Chung, Chenguang Zhu, and Michael Zeng. "SPLAT: Speech-Language Joint Pre-Training for Spoken Language Understanding." In: *arXiv preprint arXiv:2010.02295* (2020).

[56] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. "A Recurrent Latent Variable Model for Sequential Data." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28. 2015.

[57] Jill J Clark, Mary Pat Larsen, Linda L Culley, Judith Reid Graves, and Mickey S Eisenberg. "Incidence of Agonal Respirations in Sudden Cardiac Arrest." In: *Annals of Emergency Medicine* 21.12 (1992), pp. 1464–1467.

[58] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2019.

[59] Josh Coalson and Erik de Castro Lopo. *Free Lossless Audio Encoding (FLAC)*. Aug. 2019.

[60] Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. "Wav2letter: an end-to-end convnet-based speech recognition system." In: *NeurIPS Workshop: End-to-end Learning for Speech and Audio Processing* (2016).

[61] Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. "Unsupervised Cross-Lingual Representation Learning for Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2021, pp. 2426–2430.

[62] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. "Snips Voice Platform: An Embedded Spoken Language Understanding System for Private-by-Design Voice Interfaces." In: *arXiv preprint arXiv:1805.10190* (2018).

[63] D Alan Cruse, David Alan Cruse, D A Cruse, and D A Cruse. *Lexical Semantics*. Cambridge University Press, 1986.

[64] Santiago Cuervo, Maciej Grabias, Jan Chorowski, Grzegorz Ciesielski, Adrian Łańcucki, Paweł Rychlikowski, and Ricard Marxer. "Contrastive Prediction Strategies for Unsupervised Segmentation and Categorization of Phonemes and Words." In: *arXiv preprint arXiv:2110.15909* (2021).

[65] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 248–255.

[66] Li Deng, Michael L Seltzer, Dong Yu, Alex Acero, Abdel-rahman Mohamed, and Geoff Hinton. "Binary coding of speech spectrograms using a deep auto-encoder." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2010, pp. 1692–1695.

[67] Anoop Deoras, Ruhi Sarikaya, Gokhan Tur, and Dilek Hakkani-Tür. "Joint decoding for speech recognition and semantic tagging." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2012, pp. 1067–1070.

[68] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2019, pp. 4171–4186.

[69] Sander Dieleman, Charlie Nash, Jesse Engel, and Karen Simonyan. "Variable-rate discrete representation learning." In: *arXiv preprint arXiv:2103.06089* (2021).

[70] Yannis Dimopoulos, Paul Bourret, and Sovan Lek. "Use of some sensitivity criteria for choosing networks with good generalization ability." In: *Neural Processing Letters* 2.6 (1995), pp. 1–4.

[71] Laurent Dinh, David Krueger, and Yoshua Bengio. "Nice: Non-Linear Independent Components Estimation." In: *arXiv preprint arXiv:1410.8516* (2014).

[72] Carl Doersch, Abhinav Gupta, and Alexei A Efros. "Unsupervised Visual Representation Learning by Context Prediction." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1422–1430.

[73] Linhao Dong, Shuang Xu, and Bo Xu. "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition." In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 5884–5888.

[74] Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. "The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2018, pp. 1383–1392.

[75] Ewan Dunbar, Mathieu Bernard, Nicolas Hamilakis, Tu Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Eugene Kharitonov, and Emmanuel Dupoux. "The Zero Resource Speech Challenge 2021: Spoken language modelling." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[76] Ewan Dunbar, Xuan Nga Cao, Juan Benjumea, Julien Karadayi, Mathieu Bernard, Laurent Besacier, Xavier Anguera, and Emmanuel Dupoux. "The Zero Resource Speech Challenge 2017." In: *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE. 2017, pp. 323–330.

[77] Ewan Dunbar, Julien Karadayi, Mathieu Bernard, Xuan-Nga Cao, Robin Algayres, Lucas Ondel, Laurent Besacier, Sakriani Sakti, and Emmanuel Dupoux. "The Zero Resource Speech Challenge 2020: Discovering discrete subword and word units." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020, pp. 4831–4835.

[78] Ewan Dunbar et al. "The Zero Resource Speech Challenge 2019: TTS Without T." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 1088–1092.

[79] Janek Ebbers, Jahn Heymann, Lukas Drude, Thomas Glarner, Reinhold Haeb-Umbach, and Bhiksha Raj. "Hidden Markov Model Variational Autoencoder for Acoustic Unit Discovery." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2017, pp. 488–492.

[80] Vladimir Eidelman, Zhongqiang Huang, and Mary Harper. "Lessons Learned in Part-of-Speech Tagging of Conversational Speech." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2010, pp. 821–831.

[81] Lingyun Feng, Jianwei Yu, Deng Cai, Songxiang Liu, Haitao Zheng, and Yan Wang. "ASR-GLUE: A New Multi-task Benchmark for ASR-Robust Natural Language Understanding." In: *arXiv preprint arXiv:2108.13048* (2021).

[82] A. Fischer and C. Igel. "Bounding the Bias of Contrastive Divergence Learning." In: *Neural Computation* 23 (2011), pp. 664–673.

[83] A. Fischer and C. Igel. "Training Restricted Boltzmann Machines: An Introduction." In: *Pattern Recognition* 47 (2014), pp. 25–39.

[84] Ronald A Fisher. "The Use of Multiple Measurements in Taxonomic Problems." In: *Annals of Eugenics* 7.2 (1936), pp. 179–188.

[85] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. "Sequential neural models with stochastic layers." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016.

[86] Li Fu and Tinghuai Chen. "Sensitivity analysis for input vector in multilayer feedforward neural networks." In: *International Conference on Neural Networks*. IEEE. 1993, pp. 215–218.

[87] John S. Garofolo. *TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1*. Web Download. 1993.

[88] Sahar Ghannay, Antoine Caubrière, Yannick Esteve, Antoine Laurent, and Emmanuel Morin. "End-to-end named entity extraction from speech." In: *arXiv preprint arXiv:1805.12045* (2018).

[89] Thomas Glarner, Patrick Hanebrink, Janek Ebbers, and Reinhold Haeb-Umbach. "Full Bayesian Hidden Markov Model Variational Autoencoder for Acoustic Unit Discovery." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)* (2018), pp. 2688–2692.

[90] Carlos Gómez-Rodrıguez and David Vilares. "Constituent Parsing as Sequence Labeling." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2018, pp. 1314–1324.

[91] Anirudh Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. "Z-Forcing: Training Stochastic Recurrent Networks." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017.

[92] Alex Graves. "Sequence transduction with recurrent neural networks." In: *Representation Learning Workshop, ICML* (2012).

[93] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks." In: *Proceedings of the International Conference on Machine Learning (ICML)*. ACM. 2006, pp. 369–376.

[94] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al. "Conformer: Convolution-augmented Transformer for Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020.

[95] Michael Gutmann and Aapo Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models." In: *International Conference on Artificial Intelligence and Statistics (AISTATS)* (2010).

[96] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. "Learning Latent Dynamics for Planning from Pixels." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 97. PMLR, 2019, pp. 2555–2565.

[97] Kyu J Han, Seongjun Hahm, Byung-Hak Kim, Jungsuk Kim, and Ian R Lane. "Deep Learning-Based Telephony Speech Recognition in the Wild." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2017, pp. 1323–1327.

[98] Awni Hannun. "Sequence modeling with ctc." In: *Distill* 2.11 (2017), e8.

[99] Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. "First-Pass Large Vocabulary Continuous Speech Recognition Using Bidirectional Recurrent DNNs." In: *arXiv preprint arXiv:1408.2873* (2014).

[100] Jakob D. Havtorn, Jes Frellsen, Søren Hauberg, and Lars Maaløe. "Hierarchical VAEs Know What They Don't Know." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 139. PMLR, 2021, pp. 4117–4128.

[101] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[102] Michael Heck, Sakriani Sakti, and Satoshi Nakamura. "Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to zerospeech 2017." In: *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE. 2017, pp. 740–746.

[103] Charles T Hemphill, John J Godfrey, and George R Doddington. "The ATIS spoken language systems pilot corpus." In: *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*. 1990.

[104] Matthew Henderson. "Machine learning for dialog state tracking: A review." In: *Technical report* (2015).

[105] Matthew Henderson, Blaise Thomson, and Jason D Williams. "The second dialog state tracking challenge." In: *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*. 2014, pp. 263–272.

[106] Geoffrey E. Hinton. "A Practical Guide to Training Restricted Boltzmann Machines." In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller. Springer, 2012, pp. 599–619.

[107] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. "A Fast Learning Algorithm for Deep Belief Nets." In: *Neural Computation* 18.7 (2006), pp. 1527–1554.

[108] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. "Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design." en. In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2019.

[109] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory." In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[110] Nils Holzenberger, Mingxing Du, Julien Karadayi, Rachid Riad, and Emmanuel Dupoux. "Learning Word Embeddings: Unsupervised Methods for Fixed-size Representations of Variable-length Speech Segments." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2018, pp. 2683–2687.

[111] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. "Hubert: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units." In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 3451–3460.

[112] Wei-Ning Hsu, Anuroop Sriram, Alexei Baevski, Tatiana Likhomanenko, Qiantong Xu, Vineel Pratap, Jacob Kahn, Ann Lee, Ronan Collobert, Gabriel Synnaeve, et al. "Robust wav2vec 2.0: Analyzing Domain Shift in Self-Supervised Pre-Training." In: *arXiv preprint arXiv:2104.01027* (2021).

[113] Wei-Ning Hsu, Yu Zhang, and James Glass. "Learning Latent Representations for Speech Generation and Transformation." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2017, pp. 1273–1277.

[114] Wei-Ning Hsu, Yu Zhang, and James Glass. "Unsupervised learning of disentangled and interpretable representations from sequential data." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017.

[115] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. "Densely connected convolutional networks." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4700–4708.

[116] Ferenc Huszár. "Is Maximum Likelihood Useful for Representation Learning?" In: *inFERENCe* (2017). (Visited on 12/16/2021).

[117] Kathleen M Hutchinson. "Influence of sentence context on speech perception in young and older adults." In: *Journal of Gerontology* 44.2 (1989), pp. 36–44.

[118] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In: *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR. 2015, pp. 448–456.

[119] Niels Bruun Ipsen, Pierre-Alexandre Mattei, and Jes Frellsen. "not-MIWAE: Deep Generative Modelling with Missing not at Random Data." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2021.

[120] Kazuki Irie, Rohit Prabhavalkar, Anjuli Kannan, Antoine Bruguier, David Rybach, and Patrick Nguyen. "On the choice of modeling unit for sequence-to-sequence speech recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2019, pp. 3800–3804.

[121] Christiaan Jacobs, Yevgen Matusevych, and Herman Kamper. "Acoustic word embeddings for zero-resource languages using self-supervised contrastive learning and multilingual adaptation." In: *IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2021, pp. 919–926.

[122] Glorianna Jagfeld and Ngoc Thang Vu. "Encoding Word Confusion Networks with Recurrent Neural Networks for Dialog State Tracking." In: *Proceedings of the Workshop on Speech-Centric Natural Language Processing*. 2017, pp. 10–17.

[123] Eric Jang, Shixiang Gu, and Ben Poole. "Categorical Reparameterization with Gumbel-Softmax." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2017.

[124] Aren Jansen, Manoj Plakal, Ratheet Pandya, Daniel PW Ellis, Shawn Hershey, Jiayang Liu, R Channing Moore, and Rif A Saurous. "Unsupervised learning of semantic audio representations." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 126–130.

[125] Aren Jansen, Samuel Thomas, and Hynek Hermansky. "Weak top-down constraints for unsupervised acoustic model training." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2013, pp. 8091–8095.

[126] Aren Jansen and Benjamin Van Durme. "Efficient Spoken Term Discovery Using Randomized Algorithms." In: *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (2011), pp. 401–406.

[127] Arindam Jati and Panayiotis Georgiou. "Speaker2Vec: Unsupervised Learning and Adaptation of a Speaker Manifold Using Deep Neural Networks with an Evaluation on Speaker Segmentation." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2017, pp. 3567–3571.

[128] Arindam Jati and Panayiotis Georgiou. "Neural predictive coding using convolutional neural networks toward unsupervised learning of speaker characteristics." In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.10 (2019), pp. 1577–1589.

[129] Dongwei Jiang, Xiaoning Lei, Wubo Li, Ne Luo, Yuxuan Hu, Wei Zou, and Xiangang Li. "Improving Transformer-Based Speech Recognition Using Unsupervised Pre-training." In: *arXiv preprint arXiv:1910.09932* (2019).

[130] Dongwei Jiang, Wubo Li, Ruixiong Zhang, Miao Cao, Ne Luo, Yang Han, Wei Zou, Kun Han, and Xiangang Li. "A Further Study of Unsupervised Pretraining for Transformer Based Speech Recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 6538–6542.

[131] Haydn Thomas Jones and Juston Moore. "Is the Discrete VAE's Power Stuck in its Prior?" In: *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop* (2020).

[132] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. "An Introduction to Variational Methods for Graphical Models." In: *Machine Learning* 37.2 (1999), pp. 183–233.

[133] Biing-Hwang Juang and Lawrence R Rabiner. "Automatic speech recognition–a brief history of the technology development." In: *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara* 1 (2005), p. 67.

[134] Dan Jurafsky and James H Martin. "Speech and language processing. Vol. 3." In: *US: Prentice Hall* (2014).

[135] Jacob Kahn, Ann Lee, and Awni Hannun. "Self-training for end-to-end speech recognition." In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 7084–7088.

[136] Jacob Kahn, Morgane Rivière, Weiyi Zheng, Evgeny Kharitonov, Qiantong Xu, Pierre-Emmanuel Mazaré, Julien Karadayi, Vitaliy Liptchinsky, Ronan Collobert, Christian Fuegen, et al. "Libri-light: A benchmark for asr with limited or no supervision." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 7669–7673.

[137] Herman Kamper. "Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 6535–3539.

[138] Herman Kamper, Micha Elsner, Aren Jansen, and Sharon Goldwater. "Unsupervised Neural Network Based Feature Extraction Using Weak Top-Down Constraints." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2015, pp. 5818–5822.

[139] Herman Kamper and Benjamin van Niekerk. "Towards unsupervised phone and word segmentation using self-supervised vector-quantized neural networks." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2021.

[140] Herman Kamper, Weiran Wang, and Karen Livescu. "Deep Convolutional Acoustic Word Embeddings Using Word-Pair Side Information." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2016, pp. 4950–4954.

[141] Kazuya Kawakami, Luyu Wang, Chris Dyer, Phil Blunsom, and Aaron van den Oord. "Learning Robust and Multilingual Speech Representations." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 1182–1192.

[142] Sameer Khurana, Shafiq Rayhan Joty, Ahmed Ali, and James Glass. "A factorial deep markov model for unsupervised disentangled representation learning from speech." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 6540–6544.

[143] Sameer Khurana, Antoine Laurent, and James Glass. "Magic dust for cross-lingual adaptation of monolingual wav2vec-2.0." In: *arXiv preprint arXiv:2110.03560* (2021).

[144] Sameer Khurana, Antoine Laurent, Wei-Ning Hsu, Jan Chorowski, Adrian Lancucki, Ricard Marxer, and James Glass. "A Convolutional Deep Markov Model for Unsupervised Speech Representation Learning." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020, pp. 3790–3794.

[145] Chanwoo Kim, Minkyu Shin, Abhinav Garg, and Dhananjaya Gowda. "Improved vocal tract length perturbation for a state-of-the-art end-to-end speech recognition system." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2019, pp. 739–743.

[146] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2015.

[147] Diederik P Kingma and Max Welling. "Auto-Encoding Variational Bayes." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2014.

[148] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 2015.

[149] Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, and Max Welling. "Semi-Supervised Learning with Deep Generative Models." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2014.

[150] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improved variational inference with inverse autoregressive flow." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016.

[151] Durk P Kingma, Tim Salimans, and Max Welling. "Variational dropout and the local reparameterization trick." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015.

[152] Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R. Rickford, Dan Jurafsky, and Sharad Goel. "Racial disparities in automated speech recognition." In: *Proceedings of the National Academy of Sciences* 117.14 (2020), pp. 7684–7689.

[153] Mark A Kramer. "Nonlinear Principal Component Analysis Using Autoassociative Neural Networks." In: *AIChE Journal* 37.2 (1991), pp. 233–243.

[154] Felix Kreuk, Joseph Keshet, and Yossi Adi. "Self-Supervised Contrastive Learning for Unsupervised Phoneme Segmentation." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2020.

[155] Klaus Krippendorff. *Content analysis: An introduction to its methodology*. Sage publications, 2018.

[156] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images." PhD thesis. University of Toronto, 2009.

[157] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks." In: *Advances in Neural Information Processing Systems (NeurIPS)* 25 (2012).

[158] Andreas Krug and Sebastian Stober. "Introspection for convolutional automatic speech recognition." In: *EMNLP Workshop: BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. 2018.

[159] Cheng-I Lai, Yung-Sung Chuang, Hung-Yi Lee, Shang-Wen Li, and James Glass. "Semi-supervised spoken language understanding via self-supervised speech and language model pretraining." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 7468–7472.

[160] Cheng-I Jeff Lai, Yang Zhang, Alexander H Liu, Shiyu Chang, Yi-Lun Liao, Yung-Sung Chuang, Kaizhi Qian, Sameer Khurana, David Cox, and Jim Glass. "Parp: Prune, adjust and reprune for self-supervised speech recognition." In: *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021).

[161] Guokun Lai, Zihang Dai, Yiming Yang, and Shinjae Yoo. "Reexamination of the role of latent variables in sequence modeling." In: *Advances in Neural Information Processing Systems (NeurIPS)* (2019).

[162] Guokun Lai, Bohan Li, Guoqing Zheng, and Yiming Yang. *Stochastic WaveNet: A Generative Latent Variable Model for Sequential Data*. 2018.

[163] Kushal Lakhotia, Evgeny Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Adelrahman Mohamed, et al. "Generative spoken language modeling from raw audio." In: *arXiv preprint arXiv:2102.01192* (2021).

[164] Yann A. LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based Learning Applied to Document Recognition." In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2323.

[165] Chia-ying Lee and James Glass. "A nonparametric Bayesian approach to acoustic model discovery." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2012, pp. 40–49.

[166] Honglak Lee, Peter Pham, Yan Largman, and Andrew Ng. "Unsupervised feature learning for audio classification using convolutional deep belief networks." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 22. 2009.

[167] Keith Levin, Katharine Henry, Aren Jansen, and Karen Livescu. "Fixed-Dimensional Acoustic Embeddings of Variable-Length Segments in Low-Resource Settings." In: *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE. 2013, pp. 410–415.

[168] Jason Li, Vitaly Lavrukhin, Boris Ginsburg, Ryan Leary, Oleksii Kuchaiev, Jonathan M Cohen, Huyen Nguyen, and Ravi Teja Gadde. "Jasper: An End-to-End Convolutional Neural Acoustic Model." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2019, pp. 71–75.

[169] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. "Visualizing and Understanding Neural Models in NLP." In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. 2016.

[170] Jiwei Li, Will Monroe, and Dan Jurafsky. "Understanding neural networks through representation erasure." In: *arXiv preprint arXiv::1612.08220* (2016).

[171] Yingming Li, Ming Yang, and Zhongfei Mark Zhang. "A survey of multi-view representation learning." In: *IEEE Transactions on Knowledge and Data Engineering*. IEEE, 2018.

[172] Shaoshi Ling and Yuzong Liu. "Decoar 2.0: Deep contextualized acoustic representations with vector quantization." In: *arXiv preprint arXiv:2012.06659* (2020).

[173] Shaoshi Ling, Yuzong Liu, Julian Salazar, and Katrin Kirchhoff. "Deep Contextualized Acoustic Representations for Semi-Supervised Speech Recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 6429–6433.

[174] Alexander H. Liu, Yu-An Chung, and James Glass. "Non-Autoregressive Predictive Coding for Learning Speech Representations from Local Dependencies." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2021, pp. 3730–3734.

[175] Andy T Liu, Shang-Wen Li, and Hung-yi Lee. "Tera: Self-supervised learning of transformer encoder representation for speech." In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 2351–2366.

[176] Andy T Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee. "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 6419–6423.

[177] Chunxi Liu, Jan Trmal, Matthew Wiesner, Craig Harman, and Sanjeev Khudanpur. "Topic Identification for Speech without ASR." In: *arXiv preprint arXiv:1703.07476* (2017).

[178] Linda Liu, Yile Gu, Aditya Gourav, Ankur Gandhe, Shashank Kalmane, Denis Filimonov, Ariya Rastrow, and Ivan Bulyko. "Domain-aware neural language models for speech recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 7373–7377.

[179] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Attributes in the Wild." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. 2015.

[180] Bogdan Ludusan, Maarten Versteegh, Aren Jansen, Guillaume Gravier, Xuan-Nga Cao, Mark Johnson, and Emmanuel Dupoux. "Bridging the gap between speech technology and natural language processing: an evaluation toolbox for term discovery systems." In: *Proceedings of the Conference on Language Resources and Evaluation (LREC)*. 2014.

[181] Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio. "Speech Model Pre-Training for End-to-End Spoken Language Understanding." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)* (2019), pp. 814–818.

[182] Yi Luo and Nima Mesgarani. "Conv-TasNet: Surpassing Ideal Time-Frequency Magnitude Masking for Speech Separation." In: *IEEE/ACM Transactions on Audio Speech and Language* 27.8 (2019), pp. 1256–1266.

[183] Christoph Lüscher, Eugen Beck, Kazuki Irie, Markus Kitza, Wilfried Michel, Albert Zeyer, Ralf Schlüter, and Hermann Ney. "RWTH ASR systems for LibriSpeech: Hybrid vs Attention." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2019, pp. 231–235.

[184] Xuezhe Ma and Eduard Hovy. "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2016, pp. 1064–1074.

[185] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. "BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada, 2019, pp. 6548–6558.

[186] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2017.

[187] Héctor Martınez Alonso and Barbara Plank. "When is multitask learning effective? Semantic sequence prediction under varying data conditions." In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 44–53.

[188] Pierre-Alexandre Mattei and Jes Frellsen. "Leveraging the exact likelihood of deep latent variable models." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018, pp. 3859–3870.

[189] Pierre-Alexandre Mattei and Jes Frellsen. "MIWAE: Deep generative modelling and imputation of incomplete data sets." In: *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR. 2019, pp. 4413–4423.

[190] Brian McMahan and Delip Rao. "Listening to the world improves speech command recognition." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2018.

[191] Yajie Miao, Mohammad Gowayyed, and Florian Metze. "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding." In: *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE. 2015, pp. 167–174.

[192] Paul Michel, Okko Rasanen, Roland Thiollière, and Emmanuel Dupoux. "Blind Phoneme Segmentation With Temporal Prediction Errors." In: *ACL Student Research Workshop* (2017).

[193] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernock, and Sanjeev Khudanpur. "Recurrent Neural Network Based Language Model." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2010, pp. 1045–1048.

[194] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 26. 2013.

[195] Tomas Mikolov and Geoffrey Zweig. "Context dependent recurrent neural network language model." In: *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2012, pp. 234–239.

[196] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. "Efficient Estimation of Word Representations in Vector Space." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2013.

[197] Benjamin Milde and Chris Biemann. "Unspeech: Unsupervised Speech Context Embeddings." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2018, pp. 2693–2697.

[198] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. "Monte Carlo Gradient Estimation in Machine Learning." In: *Journal of Machine Learning Research* 21 (2020), pp. 1–62.

[199] Diego Mollá, Menno van Zaanen, and Steve Cassidy. "Named Entity Recognition in Question Answering of Speech Data." In: *Proceedings of the Australasian Language Technology Workshop 2007*. Dec. 2007, pp. 57–65.

[200] Edmilson Morais, Hong-Kwang J Kuo, Samuel Thomas, Zoltán Tüske, and Brian Kingsbury. "End-to-end spoken language understanding using transformer networks and self-supervised pre-trained features." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 7483–7487.

[201] Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. "Neural Belief Tracker: Data-Driven Dialogue State Tracking." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2017, pp. 1777–1788.

[202] Meinard Müller. "Dynamic Time Warping." In: *Information Retrieval for Music and Motion* (2007), pp. 69–84.

[203] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. "Multimodal deep learning." In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2011, pp. 689–696.

[204] Benjamin van Niekerk, Leanne Nortje, and Herman Kamper. "Vector-Quantized Neural Networks for Acoustic Unit Discovery in the ZeroSpeech 2020 Challenge." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)* (2020), pp. 4836–4840.

[205] Lucas Ondel, Lukáš Burget, and Jan Černock. "Variational Inference for Acoustic Unit Discovery." In: *Procedia Computer Science* 81 (2016), pp. 80–86.

[206] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. "Representation Learning with Contrastive Predictive Coding." In: *arXiv preprint arXiv:1807.03748* (2018).

[207] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. "WaveNet: A Generative Model for Raw Audio." In: *Proceedings of the 9th ISCA Speech Synthesis Workshop* (2016).

[208] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. "Conditional image generation with PixelCNN decoders." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2016, pp. 4790–4798.

[209] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel Recurrent Neural Networks." In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2016.

[210] Aäron van den Oord et al. "Parallel WaveNet: Fast high-fidelity speech synthesis." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Proceedings of Machine Learning Research. Stockholm, Sweden: PMLR, 2018, pp. 3915–3923.

[211] Yassine Ouali, Céline Hudelot, and Myriam Tami. "An Overview of Deep Semi-Supervised Learning." In: *arXiv preprint arXiv:2006.05278* (2020).

[212] Shruti Palaskar, Vikas Raunak, and Florian Metze. "Learned in speech recognition: Contextual acoustic word embeddings." In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 6530–6534.

[213] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. "Librispeech: an ASR corpus based on public domain audio books." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2015, pp. 5206–5210.

[214] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 2613–2617.

[215] Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. "Layer-wise analysis of a self-supervised speech representation model." In: *arXiv preprint arXiv:2107.04734* (2021).

[216] Ankita Pasad, Felix Wu, Suwon Shon, Karen Livescu, and Kyu J Han. "On the Use of External Data for Spoken Named Entity Recognition." In: *arXiv preprint arXiv:2112.07648* (2021).

[217] Santiago Pascual, Mirco Ravanelli, Joan Serrà, Antonio Bonafonte, and Yoshua Bengio. "Learning problem-agnostic speech representations from multiple self-supervised tasks." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 161–165.

[218] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in PyTorch." In: *NeurIPS Workshop: Autodiff* (2017).

[219] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. "Context Encoders: Feature Learning by Inpainting." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2536–2544.

[220] Douglas B. Paul and Janet M. Baker. "The Design for the Wall Street Journal-Based CSR Corpus." In: *Proceedings of the Workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[221] Karol J Piczak. "Environmental sound classification with convolutional neural networks." In: *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE. 2015, pp. 1–6.

[222] Barbara Plank, Anders Søgaard, and Yoav Goldberg. "Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. 2016, pp. 412–418.

[223] Rohit Prabhavalkar, Kanishka Rao, Tara N Sainath, Bo Li, Leif Johnson, and Navdeep Jaitly. "A Comparison of Sequence-to-Sequence Models for Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA. 2017, pp. 939–943.

[224] Vineel Pratap, Awni Hannun, Qiantong Xu, Jeff Cai, Jacob Kahn, Gabriel Synnaeve, Vitaliy Liptchinsky, and Ronan Collobert. "Wav2letter++: A fast open-source speech recognition system." In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2019, pp. 6460–6464.

[225] Yao Qian, Ximo Bianv, Yu Shi, Naoyuki Kanda, Leo Shen, Zhen Xiao, and Michael Zeng. "Speech-language pre-training for end-to-end spoken language understanding." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 7458–7462.

[226] Ashwin Ram et al. "Conversational AI: The Science Behind the Alexa Prize." In: *CoRR* abs/1801.03604 (2018).

[227] Mirco Ravanelli, Jianyuan Zhong, Santiago Pascual, Pawel Swietojanski, Joao Monteiro, Jan Trmal, and Yoshua Bengio. "Multi-task self-supervised learning for robust speech recognition." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 6989–6993.

[228] Daniel Renshaw, Herman Kamper, Aren Jansen, and Sharon Goldwater. "A Comparison of Neural Network Methods for Unsupervised Representation Learning on the Zero Resource Speech Challenge." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2015, pp. 3199–3203.

[229] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows." In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2015.

[230] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models." In: *Proceedings of the International Conference on Machine Learning (ICML)*. Vol. 32. Beijing, China: PMLR, 2014, pp. 1278–1286.

[231] Morgane Riviere, Armand Joulin, Pierre-Emmanuel Mazaré, and Emmanuel Dupoux. "Unsupervised pretraining transfers well across languages." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 7414–7418.

[232] Sara Sabour, William Chan, and Mohammad Norouzi. "Optimal completion distillation for sequence learning." In: *International Conference on Learning Representations (ICLR)*. 2019.

[233] Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. "Non-Autoregressive Machine Translation with Latent Alignments." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 1098–1108.

[234] Tara Sainath and Carolina Parada. "Convolutional Neural Networks for Small-Footprint Keyword Spotting." In: *Technical report* (2015).

[235] Justin Salamon and Juan Pablo Bello. "Deep convolutional neural networks and data augmentation for environmental sound classification." In: *IEEE Signal Processing Letters* 24.3 (2017), pp. 279–283.

[236] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. "A Dataset and Taxonomy for Urban Sound Research." In: *Proceedings of the 22nd ACM international conference on Multimedia*. ACM. 2014, pp. 1041–1044.

[237] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. "PixelCNN++: Improving the PixelCNN with Discretized Logistic Mixture Likelihood and Other Modifications." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2017.

[238] Vaibhav Saxena, Jimmy Ba, and Danijar Hafner. "Clockwork variational autoencoders." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34. 2021.

[239] Thomas Schatz, Vijayaditya Peddinti, Francis Bach, Aren Jansen, Hynek Hermansky, and Emmanuel Dupoux. "Evaluating Speech Features With the Minimal-Pair ABX Task: Analysis of the Classical MFC/PLP Pipeline." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2013.

[240] Thomas Schatz, Vijayaditya Peddinti, Xuan-Nga Cao, Francis Bach, Hynek Hermansky, and Emmanuel Dupoux. "Evaluating Speech Features With the Minimal-Pair ABX Task (II): Resistance to Noise." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2014.

[241] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. "wav2vec: Unsupervised Pre-Training for Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2019, pp. 3465–3469.

[242] Shane Settle and Karen Livescu. "Discriminative Acoustic Word Embeddings: Recurrent neural network-based approaches." In: *IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2016, pp. 503–510.

[243] Claude Elwood Shannon. "A Mathematical Theory of Communication." In: *The Bell System Technical Journal* 27.3 (1948), pp. 379–423.

[244] Bowen Shi, Wei-Ning Hsu, Kushal Lakhotia, and Abdelrahman Mohamed. "Learning audio-visual speech representation by masked multimodal cluster prediction." In: *arXiv preprint arXiv:2201.02184* (2022).

[245] Bowen Shi, Wei-Ning Hsu, and Abdelrahman Mohamed. "Robust Self-Supervised Audio-Visual Speech Recognition." In: *arXiv preprint arXiv:2201.01763* (2022).

[246] Suwon Shon, Ankita Pasad, Felix Wu, Pablo Brusco, Yoav Artzi, Karen Livescu, and Kyu J Han. "SLUE: New Benchmark Tasks for Spoken Language Understanding Evaluation on Natural Speech." In: *arXiv preprint arXiv:2111.10367* (2021).

[247] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *Proceedings of the International Conference on Learning Representations (ICLR)* (2015).

[248] Andjelic Sladjana, Panic Gordana, and Sijacki Ana. "Emergency response time after out-of-hospital cardiac arrest." In: *European Journal of Internal Medicine* 22.4 (2011), pp. 386–393.

[249] Paul Smolensky. "Chapter 6: Information Processing in Dynamical Systems: Foundations of Harmony Theory." In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. Ed. by J.L. McClelland D.E. Rumelhart. MIT Press, 1986. Chap. 6, pp. 194–281.

[250] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. "Ladder variational autoencoders." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016.

[251] Frank K Soong, AE Rosenberg, LR Rabiner, and BH Juang. "A vector quantization approach to speaker recognition." In: *ATT Tech Journal* 66 (1987), p. 22.

[252] Ajay Srinivasamurthy, Petr Motlicek, Ivan Himawan, Gyorgy Szaszak, Youssef Oualil, and Hartmut Helmke. *Semi-Supervised Learning with Semantic Knowledge Extraction for Improved Speech Recognition in Air Traffic Control*. Tech. rep. 2017.

[253] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958.

[254] Michalina Strzyz, David Vilares, and Carlos Gómez-Rodrıguez. "Viable Dependency Parsing as Sequence Labeling." In: *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. 2019, pp. 717–723.

[255] Mihai Surdeanu, Jordi Turmo, and Eli Comelles. "Named entity recognition from spontaneous open-domain speech." In: *Ninth European Conference on Speech Communication and Technology*. 2005.

[256] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9.

[257] Marco Tagliasacchi, Beat Gfeller, Félix de Chaumont Quitry, and Dominik Roblek. "Pre-training audio representations with self-supervision." In: *IEEE Signal Processing Letters* 27 (2020), pp. 600–604.

[258] Chaitanya Talnikar, Tatiana Likhomanenko, Ronan Collobert, and Gabriel Synnaeve. "Joint Masked CPC and CTC Training for ASR." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2021, pp. 3045–3049.

[259] L Theis, A van den Oord, and M Bethge. "A note on the evaluation of generative models." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2016, pp. 1–10.

[260] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. "Efficient object localization using convolutional networks." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2015, pp. 648–656.

[261] James Townsend, Thomas Bird, and David Barber. "Practical Lossless Compression With Latent Variables Using Bits Back Coding." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. 2019.

[262] Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. "Self-supervised Learning from a Multi-view Perspective." In: *Proceedings of the International Conference on Learning Representations (ICLR)* (2020).

[263] Gokhan Tur and Renato De Mori. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. John Wiley & Sons, 2011.

[264] Arash Vahdat and Jan Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020.

[265] Aaron Van Den Oord, Oriol Vinyals, et al. "Neural discrete representation learning." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017.

[266] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017.

[267] Maarten Versteegh, Roland Thiollière, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. "The Zero Resource Speech Challenge 2015." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2015, pp. 3169–3173.

[268] Andrew Viterbi. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." In: *IEEE Transactions on Information Theory* 13.2 (1967), pp. 260–269.

[269] Changhan Wang, Anne Wu, and Juan Pino. "CoVoST 2 and massively multilingual speech-to-text translation." In: *arXiv preprint arXiv:2007.10310* (2020).

[270] Chengyi Wang, Yu Wu, Yao Qian, Kenichi Kumatani, Shujie Liu, Furu Wei, Michael Zeng, and Xuedong Huang. "Unispeech: Unified speech representation learning with labeled and unlabeled data." In: *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR. 2021, pp. 10937–10947.

[271] Yu-Hsuan Wang, Cheng-Tao Chung, and Hung-yi Lee. "Gate activation signal analysis for gated recurrent neural networks and its correlation with phoneme boundaries." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2017.

[272] Jianfeng Wang and Xiaolin Hu. "Gated Recurrent Convolution Neural Network for OCR." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 334–343.

[273] Weiran Wang, Qingming Tang, and Karen Livescu. "Unsupervised Pre-Training of Bidirectional Speech Encoders via Masked Reconstruction." In: *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 6889–6893.

[274] Laurenz Wiskott and Terrence J Sejnowski. "Slow Feature Analysis: Unsupervised Learning of Invariances." In: *Neural Computation* 14.4 (2002), pp. 715–770.

[275] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. "Huggingface's transformers: State-of-the-art natural language processing." In: *arXiv preprint arXiv:1910.03771* (2019).

[276] Yuxin Wu and Kaiming He. "Group Normalization." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 3–19.

[277] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. "Achieving human parity in conversational speech recognition." In: *arXiv preprint arXiv:1610.05256* (2016).

[278] Qiantong Xu, Tatiana Likhomanenko, Jacob Kahn, Awni Hannun, Gabriel Synnaeve, and Ronan Collobert. "Iterative Pseudo-Labeling for Speech Recognition." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)* (2020), pp. 1006–1010.

[279] Yong Xu, Qiang Huang, Wenwu Wang, Philip JB Jackson, and Mark D Plumbley. "Fully DNN-based multi-label regression for audio tagging." In: *arXiv preprint arXiv:1606.07695* (2016).

[280] Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y Lin, Andy T Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, et al. "SUPERB: Speech processing Universal PERformance Benchmark." In: *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. 2021, pp. 1194–1198.

[281] Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. "Tensor Fusion Network for Multimodal Sentiment Analysis." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2017, pp. 1103–1114.

[282] Amir Zadeh, Paul Pu Liang, Louis-Philippe Morency, Soujanya Poria, Erik Cambria, and Stefan Scherer, eds. *Proceedings of Grand Challenge and Workshop on Human Multimodal Language (Challenge-HML)*. 2018.

[283] Neil Zeghidour, Qiantong Xu, Vitaliy Liptchinsky, Nicolas Usunier, Gabriel Synnaeve, and Ronan Collobert. "Fully convolutional speech recognition." In: *arXiv preprint arXiv:1812.06864* (2018).

[284] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer. 2014, pp. 818–833.

[285] Xiang Zhang, Junbo Zhao, and Yann LeCun. "Character-Level Convolutional Networks for Text Classification." In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015, pp. 649–657.

[286] Yaodong Zhang, Kiarash Adl, and James Glass. "Fast spoken query detection using lower-bound dynamic time warping on graphical processing units." In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2012, pp. 5173–5176.

[287] Yu Zhang, William Chan, and Navdeep Jaitly. "Very deep convolutional networks for end-to-end speech recognition." In: *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 4845–4849.

[288] W. Q. Zheng, J. S. Yu, and Y. X. Zou. "An experimental study of speech emotion recognition based on deep convolutional neural networks." In: *International Conference on Affective Computing and Intelligent Interaction (ACII)*. 2015, pp. 827–831.

[289] Yizhe Zhu, Martin Renqiang Min, Asim Kadav, and Hans Peter Graf. "S3VAE: Self-supervised sequential VAE for representation disentanglement and data generation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 6537–6546.