MEASURING THE IMPACT OF DOMAIN FACTORS IN SELF-SUPERVISED PRE-TRAINING

Ramon Sanabria\(^*\), Wei-Ning Hsu\(^\triangle\), Alexei Baevski\(^\triangle\), Michael Auli\(^\triangle\)

\(^*\)The University of Edinburgh
\(^\triangle\) Meta AI

ABSTRACT

Human speech data comprises a rich set of domain factors such as accent, syntactic and semantic variety, or acoustic environment. Previous work explores the effect of domain mismatch in automatic speech recognition between pre-training and fine-tuning as a whole \cite{1} but does not dissect the contribution of individual factors. In this paper, we present a controlled study to better understand the effect of such factors on the performance of pre-trained representations on automatic speech recognition. To do so, we pre-train models either on modified natural speech or synthesized audio, with a single domain factor modified, and then measure performance after fine-tuning. Results show that phonetic domain factors play an important role during pre-training while grammatical and syntactic factors are far less important. To our knowledge, this is the first study to better understand the domain characteristics of pre-trained sets in self-supervised pre-training for speech.

Index Terms speech recognition, self-supervised learning, domain mismatch

1. INTRODUCTION

Self-supervised learning has gained considerable traction in the speech community and showed consistent improvements in automatic speech recognition (ASR) tasks \cite{2, 3, 4, 5}, as well as other speech-related tasks \cite{6, 7, 8, 9}. Concretely, by pre-training a model with large amounts of unlabeled data, and then fine-tuning it on a downstream task (e.g., ASR, sentiment analysis, etc.), previous work rise the state-of-the-art by orders of magnitude.

Previous work on this topic relies on public datasets that often cover only a rather narrow domain. Consequently, recent work studied the effects of domain shift during pre-training \cite{1}. Concretely, in this work, authors study the impact of domain mismatch between the pre-training and fine-tuning sets by using data from different domains such as audiobooks \cite{10} or conversational speech \cite{11}. Among other things, the authors state that a mismatch in the domain between the pre-training and fine-tuning sets leads to a decrease in performance, as compared to a perfect domain match.

However, in the context of ASR, the term “domain” typically encompasses multiple factors, including but not limited to vocabulary, accent (e.g., prosody, syntax, and lexicon variation), and acoustics (e.g., wide and narrowband). A sparse combination of each one of these factors results on the domain definition of each dataset. For that reason, in domain-mismatch studies such as \cite{1}, where authors combine different datasets, domain factors get interlinked and are hard to quantify the effect that each individual factor has.

To measure the contribution of each domain factor, we present a controlled experiment. Specifically, we propose to automatically generate the pre-training sets altering individual domain factors such as phonotactic, lexical, syntactic, and prosody variation as well as speaker variability. We apply this variation using audio manipulation and synthesis (§3.1). After pre-training the model on the controlled pre-training set, we fine-tune it with real (human-generated) speech without any modification, and measure the impact that each factor has. It should be noted that while our generative approach does introduce some artifacts into the speech signal, this is a necessary trade-off to control each individual domain dimension – we hope that our findings are generalizable to natural data.

We perform experiments using wav2vec2.0 \cite{4}, with the 960 and 10-hour sets from the Librispeech corpus for pre-training and fine-tuning, respectively. Our experiments show that speaker variation in the pre-training data is crucial. Also, low-level domain dimensions such as phonotactics and prosody variation have a higher impact on final performance than syntactic and lexical variation. Concretely, while modifying phonotactics (without altering the phone set) hurts performance considerably, changing word order barely has an effect. We also find that large amounts of synthesized data, with enough speaker variability, can perform considerably better than pre-training on 44 hours of natural speech – which can be used to train a synthesizer in the first place.

2. DOMAIN

In speech, we define domain as a group of characteristics shared across a dataset composed by speech utterances\(^1\). We consider the following factors:

- **Acoustic Domain:** The acoustic environment where the utterance has been recorded (i.e., microphone type, acoustic conditions, signal processing drivers).
- **Syntactic Domain:** The word order that utterances follow. In other words, this aspect can be modeled with the word-level n-gram probabilities – transition probabilities between words.
- **Lexic/Semantic Domain:** The lexicon used for the speakers. For instance, some speakers may use technical vocabulary (e.g., experts on biology), while other may have more informal conversation (e.g., a day to day conversation).
- **Phonetic Domain:** In a set of speakers from the same languages, that use the same Lexic, Syntactic, and Acoustic domain, we can still find speakers with different phone sets, phonotactic patterns, prosody (i.e., rhythm, pitch, and volume fluctuation across the utterance).

In this work, for the sake of space we will focus on the latter three factors since, and leave the rest for future work.

\(^1\)We obviate timbre and other idiosyncratic speaker dimensions which can not be grouped.
3. METHODOLOGY

Our pipeline has three steps. First, we create a pre-training set with an altered domain dimension by either generating one from scratch with a speech synthesizer or by modifying the original set. Second, we pre-train the self-supervised model with the altered set. Third, we fine-tune the model on a labeled unmodified set for final evaluation. The pre-training set is composed by 960h of data (natural or synthesized) – unless specified otherwise – and we fine-tune it on the 10-hour subset of Libri-light [10]. Because we do not need to compare to previous work, we only report results on the development sets from LibriSpeech – dev-other/clean sets.

3.1. Dataset Creation

To create a controlled experimental setup, we either synthesize a modified phonetic sequence (Synthesis) or modify the original audio (Audio Modification). Both processes alter only a single dimension of the original domain of the data. In what follows, we describe each technique further.

To synthesize data, we use the publicly released weights from fairseq $^2$[12, 13, 14]. Concretely, we use their implementation of Fastspeech 2 [15] with HiFi GAN vocoder [16].

**VCTK:** We train this model on VCTK [17], which contains 44 hours of speech from 109 speakers. We will use it to measure the contribution of speaker variability during pre-training. Because VCTK is composed mostly for shorter utterances, the quality of the synthesizer trained on it degrades dramatically for long sentences. To solve this issue, we synthesize six words independently at a time and concatenate the resulting synthesized speech. This workaround produces speech of sufficient quality for our experiment setting.

**LJSpeech:** We train this model with the LJSpeech dataset [18]. LJSpeech is composed of 24 hours of speech from a single speaker. Because the utterances from this set are longer, synthesizers trained on this data model better longer dependencies and, consequently, prosody (i.e., rhythm variations at the utterance level). LJSpeech only has data from one speaker. The quality of the generated voice the quality of the speech generated is higher than VCTK’s.$^2$

To contrast our synthesis speech results, we also alter domain factors by modifying the original audio signal. We do this by segmenting the audio utterance at the phone or word level, reorder each unit, and concatenate them back. We do not apply any post-processing effects on the concatenated region to not complicate further the audio modification pipeline. This method preserves the natural variations present in human speech. However, it introduces discontinuities in the signal that generates unnatural artifacts. To detect the start and end of phone units, we use the Montreal Forced Aligner [19]. We then detect word boundaries by using the start time of the first phone and the end time of the last one.

3.2. Model, Training and Data

We use wav2vec 2.0 [4, 5] which is composed of a feature extractor and a subsequent transformer model. The feature extractor $f: \mathcal{X} \rightarrow \mathcal{Z}$ maps raw audio $\mathcal{X}$ to hidden representations $\mathcal{Z}$. This hidden representation is used as input to the transformer model $g: \mathcal{Z} \rightarrow \mathcal{C}$ that outputs a context representation $c_1, \ldots, c_r$. The model is trained to identify masked time-steps within a set of distractors. Once the model is pre-trained, we fine-tune it with Connectionist Temporal Classification [20].

|                               | scratch  | feat. pretrained |
|-------------------------------|----------|------------------|
|                               | dev-o    | dev-c            |
| pre-trained real data         | 19.3     | 13.5             |
| random initialization         | 98.3     | 98.8             |
| VCTK                          | 38.7     | 24.0             |
| LJSpeech                      | 47.1     | 28.9             |

Table 1. Librispeech dev-clean (dev-c) and dev-other (dev-o) results on systems pre-trained with data from two different synthesizers. Both synthesizers use one only speaker. We provide a topline where the system is pre-trained on the 960 hours of Librispeech (pretrained real data), and a baseline that is not pre-trained (random initialization). We also present results with a pre-trained feature extractor (feat. pretrained).

| Speakers | scratch  | feat. pretrained |
|----------|----------|------------------|
|          | dev-o    | dev-clean        |
|          | dev-o    | dev-clean        |
| 1        | 54.1     | 39.4             |
| 5        | 48.3     | 33.8             |
| 10       | 46.6     | 31.2             |
| 50       | 40.9     | 26.3             |
| 100      | 39.3     | 24.3             |

Table 2. Development results on pre-training with different speaker variability. The Speakers column designates the number of speakers used during pre-training. The data for each speaker is distributed homogeneously. We also present results with a pre-trained feature extractor (feat. pretrained).

4. RESULTS

4.1. Pre-training with Synthetic Speech

We start by analyzing if we can pre-train models on synthetic speech. To do so, we pre-train our model using data generated by our two synthesizers. Table 1 shows these results and compares them to a non-pre-trained model (random initialization) and a model pre-trained on the 960 hours of natural Librispeech data. As expected, all models pre-trained on synthetic speech perform better than the baseline, and underperform the system pre-trained on natural speech. We hypothesize that the gap with the topline model (pretrained real data) is due to the small imperfections in the human speech signal, that synthesizers cannot reproduce. Interestingly, we observe that although LJSpeech’s synthesis quality is higher $^2$, this improvement is not present in the overall WER performance. We hypothesize that this is due lack of speaker diversity.

4.1.1. Effect of Speaker Variation

In the previous section, we hypothesized that speaker diversity in the pre-training set might affect the performance of the fine-tuned model. Now we ask how important is to have speaker diversity in the pre-training set? To do so, using VCTK, we synthesize sets with different number of speakers, leaving the remaining parameters of the dataset fixed (e.g., content, utterances, length). We assign the same number of utterances to each speaker. In Table 2, we show the results of this experiment. We observe that speaker diversity is important – although it plateaus after 50 speakers. Note that even though VCTK’s synthesis quality is lower, it outperforms the model pre-trained on LJSpeech. This observation stresses the importance

$^2$Note that LJSpeech’s mean opinion score is considerably higher than VCTK’s (see Table 1, and Table 3 in Wang et al.[13])
Table 3. Results on Semantic domain. We synthesize the transcriptions of Librispeech, Switchboard, and Switchboard+Fisher using VCTK (only one speaker), pre-train models on this data, fine-tune using LL-10h and measure performance on Librispeech dev-other/dev-clean. All models use a pre-trained feature extractor.

| Corpus            | dev-other | dev-clean |
|-------------------|-----------|-----------|
| Librispeech       | 37.1      | 22.8      |
| Switchboard       | 39.7      | 25.3      |
| Switchboard + Fisher | 39.7  | 25.0      |

Table 4. Results on shuffling the order of words either by modifying the original audio (mod), or synthesizing a sequence of shuffled words (synth). word random shuffled uses the same number of segments and duration as word shuffled but with a random start time. Synth, and mod baselines are systems pre-trained on the unmodified synthesized and audio pre-training sets.

| Type              | dev-other | dev-clean |
|-------------------|-----------|-----------|
| Baseline          |           |           |
| Word shuffled     | 35.5      | 22.7      |
| Synth             | 36.5      | 23.0      |
| Word shuffled     | 19.8      | 13.9      |
| Mod               | 23.9      | 16.5      |
| Word random shuffled | 30.5  | 22.7      |

Table 5. Results on phone shuffling during pre-training on synthesized (synth), and the original audio (mod) using a pre-trained feature extractor. We compare these results with a non-pretrained baseline (no pre-training), and with system pre-trained on the 960 hour set from Librispeech (baseline).

| Type              | dev-other | dev-clean |
|-------------------|-----------|-----------|
| Baseline          |           |           |
| Synth             | 35.5      | 22.7      |
| Phone shuffled    | 46.6      | 30.9      |
| Phone random shuffled | 39.5  | 29.2      |
| No pre-training   | 68.8      | 57.9      |

Fig. 1. Illustration of wrd rnd (bottom) using ground truth segments (up) shuffled.
We present a controlled study to understand how individual domain factors affect self-supervised speech representation learning. With the help of speech synthesis and audio modification we alter specific domain dimensions of the pre-training set and measure performance on ASR. We observe that phonetic domain is much more important than altering the syntactic and semantic domain. Initial experiments indicate that wav2vec2.0 can be pre-trained on sequences of synthetic noise sequences that does not share any phonetic, semantic, syntactic, phonotactic, or lexical, it still outperforms a non-pre-trained baseline. This result is consistent with previous work on NLP that successfully pre-train Transformer models with out-of-domain [23] or pseudo-random [24] data. As random baselines, we experiment with pre-training the model on fewer structure data. Concretely, we experiment using Gaussian noise (white noise), and sequences of tones and noises similar to the synthetic languages, but instead of using phone units, we randomly sample with a duration of 300ms with a ±30ms sampled variation (word noise seq.) – the average duration of a word. Results show that when pre-training on these sets the system either underperforms the baseline or can not be trained.

4.5. Large Synthetic Set vs. Small Real Set

Ultimately, as a supplementary experiment, we compare the performance of pre-training on a large synthesized pre-training set and the real data used to train the synthesizer. Concretely, we synthesize all Librispeech using VCTK or Librispeech (a total 960 hours of synthesized speech with the available speakers in each synthesizer), and compare it with a model pre-trained on the training set for each synthesizer (24, and 44 hours respectively). Table 8 show the results of this comparison. We observe that the large-synthesized sets, outperform the smaller ones with real speech – specially for VCTK. This result suggests that when only few data is available for pre-training, one could use a synthesizer to augment the amount of data to improve results.

5. CONCLUSIONS AND FUTURE WORK

We present a controlled study to understand how individual domain factors affect self-supervised speech representation learning. With the help of speech synthesis and audio modification we alter specific domain dimensions of the pre-training set and measure performance on ASR. We observe that phonetic domain is much more important than altering the syntactic and semantic domain. Initial experiments indicate that wav2vec2.0 can be pre-trained on sequences of synthetic noise sequences that does not share any phonetic, semantic nor syntactic similarity with the target data.

In terms of future work, we plan to experiment with other architectures such as HuBERT [25], and confirm our results with human generated data.

Table 6. Effect of prosody variation of speech synthesized with the Librispeech model. The first column determines the number of words synthesized in isolation before concatenation.Feat. pretrained indicates that we used features from a fully pretrained.

| words | dev-other | dev-clean | feat. pretrained | dev-other | dev-clean |
|-------|-----------|-----------|------------------|-----------|-----------|
| 2     | 50.8      | 34.2      | 42.2             | 26.2      |
| 3     | 48.9      | 32.2      | 36.2             | 21.6      |
| 5     | 49.9      | 32.4      | 36.4             | 20.5      |
| no seg.| 47.1      | 28.9      | 35.5             | 20.2      |

Table 8. Comparison of models pretrained on synthesized LibriSpeech (synth) and real speech data used to train the synthesizer (human). The Source column specify how the data has been generated. The data column which pre-training set is used. For instance the first row a synthesizer trained on VCTK generates the Librispeech data set. The VCTK model is pretrained with 109 speakers. Feat. pretrained indicates that we used features from a fully pretrained.

| source | data | scratch | feat. pretrained | dev-o | dev-c |
|--------|------|---------|------------------|-------|-------|
| synth (VCTK) | LS | 38.7 | 31.8 | 20.1 |
| synth (LJ) | LS | 47.1 | 35.5 | 20.2 |
| human   | VCTK | 49.7 | 36.2 | 26.1 |
| human   | LJ  | 49.8 | 33.1 | 22.7 |

Table 7. Results on pre-training with the random tone and noise sequences of phone length (phone tone seq) and word length (word noise seq.). We compare these systems to a system trained on white noise (white noise), and a system without pre-training (no pre-training). The symbol ‘-’ designates that the model did not convergence. Feat. pretrained indicates that we used features from a fully pre-trained.

| method   | scratch | feat. pretrained |
|----------|---------|------------------|
| no pre-training | 98.3 | 68.8 |
| white noise    | -     | 71.2 |
| word noise seq.| -     | 59.4 |
| phone tone seq.| -     | 47.6 |
6. REFERENCES

[1] W.-N. Hsu, A. Sriram, A. Baevski, T. Likhomanenko, Q. Xu, V. Pratap, J. Kahn, A. Lee, R. Collobert, G. Synnaeve et al., “Robust wav2vec 2.0: Analyzing domain shift in self-supervised pre-training,” in Interspeech, 2021.

[2] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, “An unsupervised autoregressive model for speech representation learning,” in Interspeech, 2019.

[3] S. Schneider, A. Baevski, R. Collobert, and M. Auli, “wav2vec: Unsupervised pre-training for speech recognition,” in Interspeech, 2019.

[4] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in NeurIPS, 2020.

[5] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, “Unsupervised cross-lingual representation learning for speech recognition,” in Interspeech, 2020.

[6] S.-W. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhota, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin et al., “Superb: Speech processing universal performance benchmark,” in Interspeech, 2021.

[7] A. Polyak, Y. Adi, J. Copet, E. Khartonov, K. Lakhota, W.-N. Hsu, A. Mohamed, and E. Dupoux, “Speech resynthesis from discrete disentangled self-supervised representations,” in Interspeech, 2021.

[8] A. Lee, H. Gong, P.-A. Duquenne, H. Schwenk, P.-J. Chen, C. Wang, S. Popuri, J. Pino, J. Gu, and W.-N. Hsu, “Textless speech-to-speech translation on real data,” arXiv preprint arXiv:2112.08352, 2021.

[9] A. Lee, P.-J. Chen, C. Wang, J. Gu, X. Ma, A. Polyak, Y. Adi, Q. He, Y. Tang, J. Pino et al., “Direct speech-to-speech translation with discrete units,” arXiv preprint arXiv:2107.05604, 2021.

[10] J. Kahn, M. Rivière, W. Zheng, E. Khartonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen et al., “Libri-light: A benchmark for asr with limited or no supervision,” in ICASSP, 2020.

[11] J. J. Godfrey, E. C. Holliman, and J. McDaniel, “Switchboard: Telephone speech corpus for research and development,” in Acoustics, Speech, and Signal Processing, IEEE International Conference on, 1992.

[12] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, “fairseq: A fast, extensible toolkit for sequence modeling,” in Proc. of NAACL System Demonstrations, 2019.

[13] C. Wang, W.-N. Hsu, Y. Adi, A. Polyak, A. Lee, P.-J. Chen, J. Gu, and J. Pino, “fairseq s2: A scalable and integrable speech synthesis toolkit,” in EMNLP: System Demonstrations, 2021.

[14] “Weights used for synthesize speech.” https://github.com/pytorch/fairseq/tree/main/examples/speech_synthesis, 2021.

[15] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, “Fastspeech 2: Fast and high-quality end-to-end text to speech,” in ICLR, 2021.

[16] J. Kong, J. Kim, and J. Bae, “HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis,” in NeurIPS, 2020.

[17] “CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit,” 2017.

[18] K. Ito and L. Johnson, “The LJ speech dataset,” https://keithito.com/LJ-Speech-Dataset/, 2017.

[19] “Montreal Forced Aligner,” https://github.com/MontrealCorpusTools/Montreal-Forced-Aligner.

[20] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in ICML, 2006.

[21] D. Berrebhi, R. Collobert, N. Jaitly, and T. Likhomanenko, “More speaking or more speakers?” ICASSP, 2023.

[22] C. Cieri, D. Miller, and K. Walker, “The fisher corpus: A resource for the next generations of speech-to-text.” in LREC, 2004.

[23] I. Papadimitriou and D. Jurański, “Learning Music Helps You Read: Using transfer to study linguistic structure in language models,” in EMNLP. Association for Computational Linguistics, 2021.

[24] K. Krishna, J. Bigham, and Z. C. Lipton, “Does pretraining for summarization require knowledge transfer?” in EMNLP, 2021.

[25] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhota, R. Salakhutdinov, and A. Mohamed, “Hubert: Self-supervised speech representation learning by masked prediction of hidden units,” in Transactions on Audio Speech and Language Processing (TASLP), 2021.