Speech & Speaker Recognition Project 2021
Semi-supervised training of a deep neural network for phoneme recognition

Alexander Tuoma
KTH Royal Institute of Technology
tuoma@kth.se

Beatrice Lovely
KTH Royal Institute of Technology
bhallm@kth.se

Georgios Moschovis
KTH Royal Institute of Technology
geomos@kth.se

Eirini Stratigi
KTH Royal Institute of Technology
stratigi@kth.se

Abstract

Semi-supervised learning (SSL) leverages a mixture of a small amount of labeled data and a larger set of unlabeled data to maintain good accuracy with much less labeled data. In this project, we re-implement the algorithm Mean Teacher [22] in a Long-Short Term Memory network architecture. Mean Teacher a semi-supervised method for consistency-based regularization. We compare performance against a baseline model with the same architecture, trained on the full labeled TIMIT training set (minus validation data) and varying amounts of labeled data. We show that Mean Teacher achieves a higher accuracy compared to the baseline model for phoneme classification on the TIMIT database with varying amount of unlabeled data. For example, with 10% labeled samples Mean Teacher achieves 52.76% accuracy compared to the baseline model that achieves 50.03% accuracy. The Mean Teacher model performs worse than the baseline model trained fully supervised, but a small reduction in accuracy is to be expected with significantly less labeled data. We expect the performance of Mean Teacher to improve with further hyperparameter tuning . All code used in our experiments is available on GitHub. [1]

1 Introduction

Speech recognition devices are becoming very common in everything from home speaker systems to customer services and even communicating with robots. Spoken language is the most natural form of communication for humans yet it is very difficult for machines to understand due to the great variety of languages, accents and voice patterns that are as unique as a fingerprint.

Training a deep neural network to recognize speech is a difficult task that typically requires a great deal of data. Yet labeling many hours of speech is a very time-consuming and expensive process, this is the main reason for training machine learning models in a semi-supervised fashion. [18] [8] [20]

In this project, we re-implemented the Mean Teacher [22], a semi-supervised learning algorithm, on top of a Long-Short Term Memory (LSTM) network to perform frame-based phoneme-classification on the TIMIT dataset[2]. We test the model on different amounts of labeled training data and compare

[1] https://github.com/BeaLove/SemiSupervisedLearningASR.
[2] TIMIT dataset is publically available at https://catalog.ldc.upenn.edu/LDC93S1.

its performance to the same network trained without semi-supervised learning on the same subsets of data as well as the full set of labeled training data. We also compare performance with [8]. The Mean Teacher method is described in more detail in subsection 4.3.

2 Related work

2.1 Semi-Supervised Learning approaches

There are two main strategies for semi-supervised learning: *self-training* and *consistency regularization*. The first approach uses a "teacher" network trained on a small amount of labeled data to generate pseudo-labels [15] for unlabeled data. A "student network" is then trained on a mix of the labeled training data and the pseudo-labeled data. [18], [20] and [8] use this method. The student network will typically perform better than the teacher network since it was trained on more data.

The *consistency regularization* approach as used in [22] and [13], uses a consistency loss term to force the student and teacher networks to make similar predictions for similar data points. Consistency regularization also forces the network to be insensitive to perturbations (noise) in input data. Mean Teacher is described further in subsection 4.3Data augmentation is another example of consistency regularization. Data augmentation is applied to speech data in [17] by masking frequencies and time-warping input sequences to introduce noise.

The Mean Teacher algorithm was originally developed and tested on image data [22], however it has been applied to speech data for sound event detection and classification in an attention convolutional recurrent neural network (CRNN) [12].

An alternative to semi-supervised approaches is self-supervision based approaches such as Wav2vec [3]. This approach enables a network to pre-train on entirely unlabeled datasets by learning to distinguish a perturbed sample from an original unchanged sample, followed by fine-tuning on labeled data. Using only 1 hr of labeled data, wav2vec outperforms the state-of-the-art on 100x more labeled data (100 hrs) [3]. With only 10 minutes of labeled data, pre-training on 53,000 hrs of unlabeled data, this model achieves 4.3 and 8.2 word error rates on clean and non-cleaned data respectively.

2.2 Speech recognition with neural networks

Probability based HMM models [16] have historically performed well on speech recognition tasks but they have been outperformed by neural network models such as encoder-decoder methods [4] and recurrent models [9] [18]. Convolutional Neural Networks, normally reserved for image tasks, have also been successfully applied to speech recognition tasks by making use of the raw speech signal spectrogram. Such methods have achieved results on par with methods based on traditional liftered mel-frequency cepstrum coefficient extraction. [19] [1]

In the last few years, state-of-the-art performance has been achieved by attention-augmented models such as transformers [5] and attention-augmented convolutional recurrent networks [21]. Transformers have proved to be even better equipped to model long-term dependencies in speech and text by applying a dot-product of an attention vector to encoded input and decoding using a context vector. This allows the network to learn to "pay attention" to the most important aspects of data over longer time-periods. [24].

3 Dataset and Preprocessing

In this project the TIMIT dataset was used. This consists of a total of 6300 recordings of different sentences spoken by 10 different speakers, both men and women with different accents of American English. These recordings are divided into a train set of 4620 samples and a test set of 1680 samples and from the former set, we created a validation set.

We used 5% of the train set for validation to tune hyperparameters and the remaining 95% as a training set and tested performance on the full test set as recommended in the TIMIT documentation. As the dataset is rather small, we, unfortunately, could not afford a larger validation set to facilitate regularization methods such as early stopping. The gain from training with more data, both in terms of lower test loss and higher test accuracy, is comparably higher than the gain from better adjusted hyperparameter values.

Preprocessing consisted of enframing with a frame size of 32 ms and a step size of 16 ms, windowing and applying a preemphasis filter. We extracted 13 coefficients of the Mel Cepstrum MFCC features. Each frame was labeled according to the phoneme that appears in it, based on the phoneme transcription in TIMIT. We reduced the 61 phonemes that appear in TIMIT to 50 according to the table of groupings that appear in [16]. These groupings allow for simplifying the output space of the model by allowing for mistakes among phonemes that sound very similar. Preprocessing of both the data, i.e. MFCC coefficients, and the targets (phoneme labels per frame) take places in a specially defined dataloader.

4 Methods

4.1 Mel Frequency Cepstrum Coefficient (MFCC)

The Mel Frequency Cepstrum Coefficient (MFCC) is a method of feature extraction of voice signals. Feature extraction is the process of determining a value or vector that can be used to identify an object or data sample. MFCC is the most widely used representation in speech processing, because the feature vectors are largely uncorrelated and thus easy to work with and capture the necessary information well. [2] The following figure represents the steps needed to extract all the necessary information:

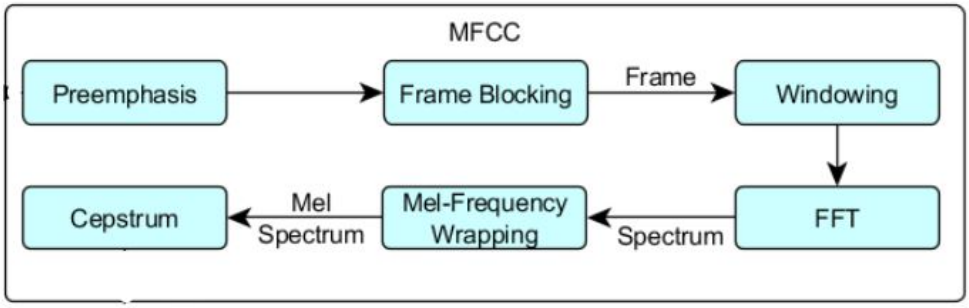


Figure 1: A visual illustration of the data preprocessing steps from [2]

4.2 Long Short Term Memory Network

We designed an architecture of Long Short Term Memory (LSTM) units that is a reduced complexity version of [18]. The output layer is a fully-connected linear layer. We use dropout and early stopping to prevent over-fitting.

LSTMs are a special class of recurrent neural networks that are capable of learning from long-term dependencies in sequenced data. In a recurrent neural network, the output from the previous layer is given as input along with a new sample in each time step, capturing dependencies in time. This allows the model to "remember" past information. LSTMs are typically even better than normal RNNs or HMMs at remembering long-term information. This special capability of LSTMs lies in its three "gates":

1. The **forget gate** determines how much of the information currently contained in the unit should be remembered. This allows an LSTM to learn longer time dependencies by discard-

ing (forgetting) some information that is less important, by controlling the extent to which a value remains in the cell.

2. The **input gate** learns to quantify the importance of new input information and decides which to keep by controlling the extent to which a new value flows into the cell.
3. The **output gate** decides which information should be passed on to the next time step (this is the "recurrent" part) by controlling the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

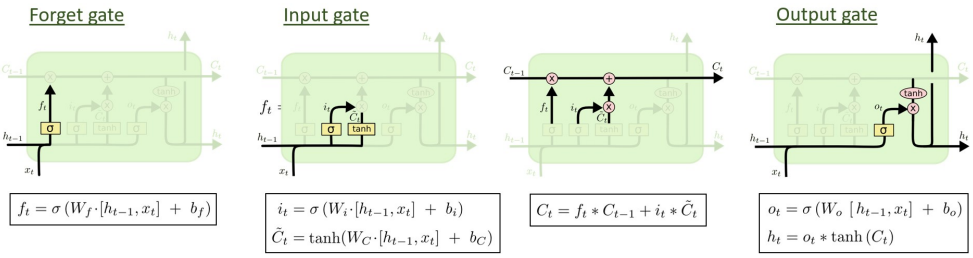


Figure 2: A visual illustration of an **LSTM Network** and the mathematical operation behind its gates, figure from `https://colah.github.io/posts/2015-08-Understanding-LSTMs/` [6], slightly modified

These three gates, see Figure 2 regulate which information is saved in the cell state - the internal memory of the LSTM unit. An LSTM also has a hidden state, similar to HMMs, this can be viewed as the working memory of the LSTM unit. For a detailed description of the functions of an LSTM see [10] or the blog [6].

4.3 Mean Teacher

The Mean Teacher algorithm was proposed in 2017 by Tarvaninen and Valpola [22]. It is a semi-supervised algorithm proposed as an alternative to the very effective but computationally heavy Temporal Ensemble Learning method which employs a temporal averaging, or consensus of predictions rather than weights [13]. In this algorithm, both the student and teacher networks are trained at the same time, on a mix of labeled and unlabeled data. For labeled samples the loss term is the sum of the cross entropy loss between prediction and target and consistency loss between models. For unlabeled samples only the consistency loss is applied. The consistency loss forces the student and teacher networks to give similar output for similar data.

The student and the teacher network are fed with samples that are similar but not identical. To achieve this, gaussian noise or other data augmentation techniques are applied to a sound sample, and the "clean" sample is fed to one network while the augmented sample is given to the other. [22] Backpropagation is only applied to the student network, while the weights of the teacher network are updated as an exponential moving average (EMA) of the weights of the student network (hence Mean Teacher). The loss function is defined as:

$$\mathcal{L}(\theta)_{\text{class}} = \frac{1}{n} \sum_{i=1}^{n} \text{MSE}(s_i, y_i)$$

$$\mathcal{L}(\theta)_{\text{consistency}} = \frac{1}{n + n\prime} \sum_{i=1}^{n,n\prime} \text{CE}(s_i, t_i)$$

$$\mathcal{L}(\theta) = \mathcal{L}(\theta)_{\text{class}} + \gamma(t)\mathcal{L}(\theta)_{\text{consistency}}$$

where n and n' are the number of labeled and unlabeled samples. s_i is the prediction from the student model and t_i is the prediction from the teacher model. $\gamma(t)$ is the consistency weight which is updated every epoch using a sigmoid rampup. For the class loss, a standard cross-entropy (CE) loss is computed between the predicted label s_i from the student model and the ground truth target y_i. For the consistency loss, mean squared error (MSE) is computed between the output of the student model

s_i and the teacher model t_i for each sample i. This encourages the two networks to give consistent output for similar data [22]. According to the authors the two hyperparameters consistency weight and EMA delay (α) are sensitive and need to be tuned to not degrade the performance. The update formula for the parameters of the teacher model θ' (in blue) given the weights of the student model θ (in orange) is as follows:

$$\theta_t\prime = \alpha\theta_{t-1}\prime + (1-\alpha)\theta_t, \alpha \in \mathbb{R}$$

4.4 Experimental Setup

As a baseline comparison model we trained our network on the full set of labeled training samples in TIMIT (minus 5% for validation). We then apply the Mean Teacher [22] semi-supervised learning algorithm with different proportions of labeled and unlabelled data and compare its accuracy to the baseline model trained on the full training set and on subsets of the training set corresponding to the amounts of labeled data given to the Mean Teacher algorithm.

We experimented with several model architectures and regularization techniques to determine the best performing baseline model, with the classic Adam optimizer and a variation of it with normalized gradients as introduced in [23]. We experimented with the number of layers, LSTM units per layer, learning rate, and dropout. Initial training was performed for 200 epochs, using the separate hold-out set to tune network hyperparameters. We found a sweet spot at 60 epochs.

4.5 Practical Setup and Hardware

We used PyTorch Kaldi for preprocessing the data and implemented the algorithms and models from scratch with PyTorch. Network training was performed on different types of GPUs both in the cloud and on a laptop.

5 Results

In this section, the results are presented.

The best accuracy score achieved was 60.1%: for the baseline model trained fully supervised (the full labeled TIMIT training set), as shown in Table 1. This was with 3 layers of LSTMs with 100 nodes in each layer and dropout 0.1. Table 1 details the performance of both the Mean Teacher and baseline models on different amounts of labeled data. The Mean Teacher method outperforms the baseline model for subsets of labeled data. Mean Teacher performs best with 30% labeled data, which brings it within 3% of the baseline performance on the full training set.

Model/Labeled data :	10%	20%	30%	100%
Mean Teacher	**52.76%**	**55.50%**	**57.28%**	-
Baseline:	50.03%	52.93%	55.26%	60.1%

Table 1: Accuracy of the best performing architecture: 3 layers with 75 nodes each, trained for 60 epochs

Table 2 shows the performance of both models with 2 layers, 75 nodes, and dropout 0.1 trained for 60 epochs.

Model/Labeled Data :	10%	20%	30%	100%
Mean Teacher:	**51.76%**	**54.24%**	**56.03%**	-
Baseline:	47.78%	51.00%	53.74%	58.69%

Table 2: Accuracy of both models with 2 layers and 75 nodes each, dropout 0.1, trained for 60 epochs

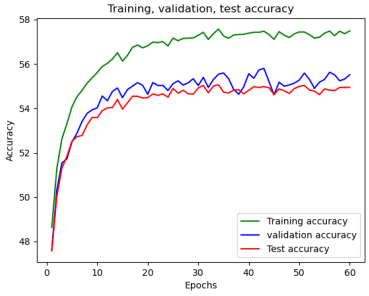
Table 3 shows both models performance with 1 layer and 100 nodes. We trained these models for 100 epochs, however the baseline model with the smaller subsets of labeled data was stopped at 34

epochs as it begins to overfit. This is unsurprising given the small amount of data. See **??** and **??** for a visual representation of the accuracy and loss evolution over epochs for the one layer model.

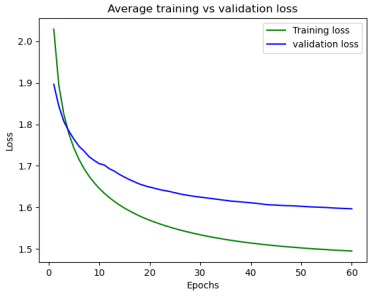
We noticed that the gradient normalization trick introduced in [23], provides an increase in accuracy when performing fully supervised learning for a large number of nodes such as 100, but it does not have a positive influence on accuracy when using fewer nodes or less data. We illustrate the positive effect of gradient normalization in Figure 3, compared to the same architecture without gradient normalization, Figure 4. Figure 5 provides a visual representation of the different network's performances with different percentages of labeled data.

Model/Labeled Data :	10%	20%	30%	100%
Mean Teacher:	**45.95%**	**48.78%**	**50.27%**	-
Baseline:	45.59%	48.37%	50.03%	55.06%

Table 3: Accuracy of both models with 1 layer, 100 nodes, trained for 100 epochs



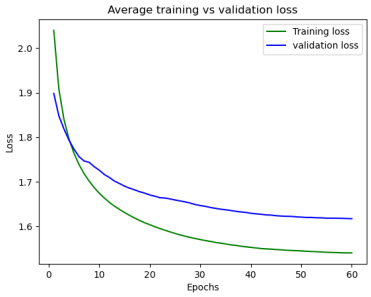
(a) Accuracy on full dataset, 1 layer, 100 nodes 60 epochs with gradient normalization



(b) Loss on full dataset, 1 layer, 100 nodes 60 epochs with gradient normalization

Figure 3: Accuracy and loss for the 1 layer, 100 node architecture with Gradient Normalization

(a) Accuracy on full dataset, 1 layer, 100 nodes 60 epochs without gradient normalization



(b) Loss on full dataset, 1 layer, 100 nodes 60 epochs without gradient normalization

Figure 4: Accuracy and loss for the 1 layer, 100 node architecture without Gradient Normalization

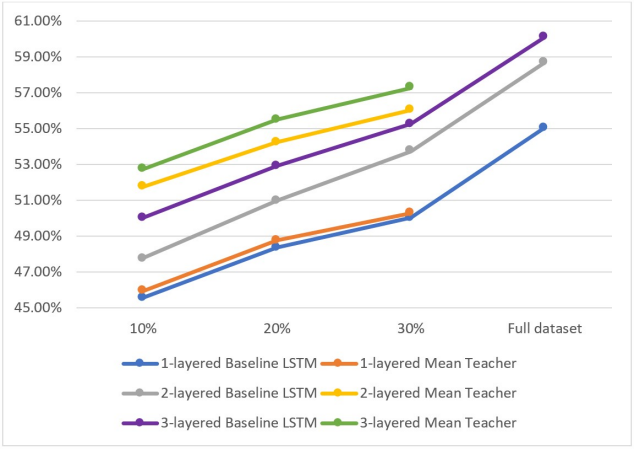


Figure 5: Test accuracies of all the models included in our experiments as reported in Table 1, Table 2, Table 3

6 Discussion and Conclusions

As expected, we see the best performance using the full labeled TIMIT training set see Table 1, however, the Mean Teacher SSL model comes close in performance, as was our intention to show.

Overall, our results show that semi-supervised learning on small subsets of labeled data combined with unlabeled data is a feasible and useful alternative to the time-consuming task of fully annotating large sets of speech data. This has been observed many times in past research [18] [8] [3]. We fully expect that with more careful tuning of hyperparameters, we would be able to further improve the performance of the Mean Teacher algorithm. Tarvainen and Valpola [22] observe that the Mean Teacher algorithm is sensitive to some hyperparameter settings, specifically the EMA decay and consistency weight [22]. See subsection 4.3.

With our best performing architecture, the Mean Teacher model performs (with different amounts of labeled data) between 3-7% worse than the baseline model as trained on the full TIMIT training set (minus validation data), see Table 1. Naturally, using the least amount of labeled data possible is the cheapest from a data perspective. These reductions in performance with less data are consistent with the observations of [8]. The Mean Teacher always outperforms the baseline model trained on the same amounts of labeled data (with no unlabeled samples), by between 2-4%. Interestingly, the best absolute improvement in accuracy (4%) was the Mean Teacher model with 2 layers of 75 nodes each trained on only 10% labeled data, as compared to the same architecture baseline model on the same subset of labeled data (Table 2). Clearly a more complex model is beneficial as compared to a single layer (Table 3)

As a state-of-the-art comparison, we compare the performance of our best model with that of Dhaka and Salvi [8]. They used the TIMIT dataset, with an auto-encoder based neural network architecture for semi-supervised learning. With 30% labeled data [8] achieve a test error up to 68.83% while our best performing Mean Teacher model achieves up to 57 % test accuracy, see Table 4.

Model/Labeled data :	20%	30%
Mean Teacher	55.50	57.28
Neural Network SOTA	**67.80**	**68.83**

Table 4: Comparison between our Mean Teacher model and [8]

The performance of even our baseline model with 100% of training data is not particularly good (60.1% accuracy), see Table 1, well below that of [8] and [18]. LSTM and RNN models, in general, are notoriously difficult and time-consuming to train [7], so it is reasonable to suggest that with more training time and very careful tuning of hyperparameters we would have been able to achieve better accuracy on the baseline and adapt the same architecture for a Mean Teacher. Our dataset

was also very limited in size. The article [7] suggests it is time to fully replace LSTMs and RNNs with attention based models but we think it is reasonable to continue experimenting with different architectures, LSTMs included.

However, some of the best performance to date in semi-supervised speech recognition to date appears to be wav2vec[3]. Wav2vec is actually self-supervised and uses as little as 10 minutes of labeled data, as described in section 2. Hari et al. [18] also achieve WERR (word error rate reductions) of 10-20% compared to their baseline model in their experiment with training on massive unlabeled datasets. Wav2vec [3] and [18] however train on significantly larger unlabeled datasets (upwards of 54,000 hrs and 1 million hours respectively) than us and [8].

6.1 Future work

It would be very interesting for the future to experiment further with Mean Teacher and other semi-supervised learning algorithms on speech data with a larger dataset and very careful hyperparameter tuning and compare their performance to a pseudo-labeling system, similar to the one that was used in [18] and [8].

Furthermore, we strongly suggest that future work investigate the choice of architecture as we mostly focused on tuning the hyperparameter values such as the number and size of hidden layers. One possible improvement would be to use bi-directional LSTMs (bi-LSTMs), similar to those illustrated in Figure 6 below, as well as combining several of those in deeper architectures. We would further recommend combining multiple bi-LSTMs and deep bi-LSTMs in an Encoder-Decoder architecture such as [4] to investigate whether performance in terms of accuracy can be improved.

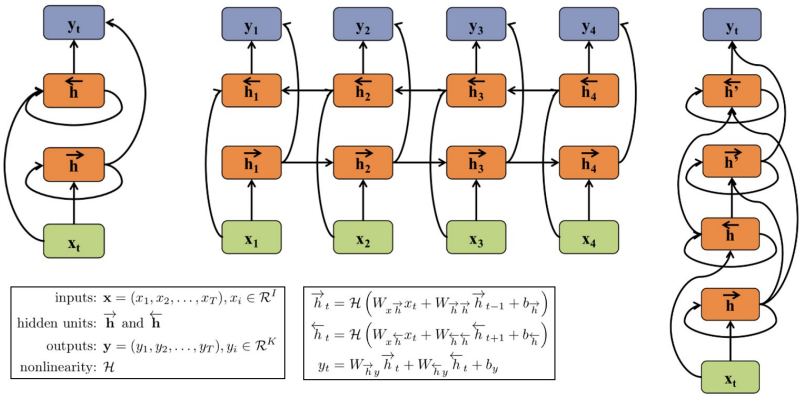


Figure 6: A visual illustration of a **bi-directional RNN**, its unfolded representation and the relevant mathematical operations, as well as a **deep bi-directional RNN**, figure from Lecture 7 slides in `http://www.cs.cmu.edu/~ytsvetko/courses/machine-learning/slides/`, slightly modified

In addition, we would like to further investigate different methods of data preprocessing, such as using dynamic feature extraction to enrich the MFCC coefficients with additional information, as well as using different architectures than the Mean Teacher. In this direction, methods that have been used in Computer Vision could potentially be adjusted to speech data, including FixMatch [20] and Temporal Ensembling, including the Π-model variation [14]. The student-teacher model used in music information retrieva; could also be tested [11]. Last but not least, transformer networks may also serve as an alternative [5].

References

[1] Ossama Abdel-Hamid et al. "Convolutional Neural Networks for Speech Recognition". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22.10 (2014), pp. 1533–1545. DOI: 10.1109/TASLP.2014.2339736.

[2] D. Anggraeni et al. "The Implementation of Speech Recognition using Mel-Frequency Cepstrum Coefficients (MFCC) and Support Vector Machine (SVM) method based on Python to Control Robot Arm". In: *IOP Conference Series: Materials Science and Engineering* 288 (Jan. 2018), p. 012042. DOI: 10.1088/1757-899x/288/1/012042. URL: https://doi.org/10.1088/1757-899x/288/1/012042.

[3] Alexei Baevski et al. "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations". In: *CoRR* abs/2006.11477 (2020). arXiv: 2006.11477. URL: https://arxiv.org/abs/2006.11477.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL].

[5] Xuankai Chang et al. *End-to-End Multi-speaker Speech Recognition with Transformer*. 2020. arXiv: 2002.03921 [eess.AS].

[6] Christopher Colah. *Understanding LSTM Networks*. URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[7] Eugenio Culurciello. *The fall of RNN / LSTM*. URL: https://towardsdatascience.com/the-fall-of-rnn-lstm-2d1594c74ce0.

[8] Akash Dhaka and Giampiero Salvi. "Sparse Autoencoder Based Semi-Supervised Learning for Phone Classification with Limited Annotations". In: Aug. 2017, pp. 22–26. DOI: 10.21437/GLU.2017-5.

[9] Alex Graves and Navdeep Jaitly. "Towards End-to-End Speech Recognition with Recurrent Neural Networks". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. ICML'14. Beijing, China: JMLR.org, 2014, II–1764–II–1772.

[10] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[11] Sangeun Kum et al. *Semi-supervised learning using teacher-student models for vocal melody extraction*. 2020. arXiv: 2008.06358 [eess.AS].

[12] Jin-Yeol Kwak and Yong-Joo Chung. "Sound Event Detection Using Derivative Features in Deep Neural Networks". In: *Applied Sciences* 10.14 (2020). ISSN: 2076-3417. DOI: 10.3390/app10144911. URL: https://www.mdpi.com/2076-3417/10/14/4911.

[13] Samuli Laine and Timo Aila. "Temporal Ensembling for Semi-Supervised Learning". In: *CoRR* abs/1610.02242 (2016). arXiv: 1610.02242. URL: http://arxiv.org/abs/1610.02242.

[14] Samuli Laine and Timo Aila. *Temporal Ensembling for Semi-Supervised Learning*. 2017. arXiv: 1610.02242 [cs.NE].

[15] Dong-Hyun Lee. "The simple and efficient semi-supervised learning method for deep neuralnetworks". In: (2013). URL: https://www.researchgate.net/publication/280581078_Pseudo-Label_The_Simple_and_Efficient_Semi-Supervised_Learning_Method_for_Deep_Neural_Networks.

[16] K.-F. Lee and H.-W. Hon. "Speaker-independent phone recognition using hidden Markov models". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.11 (1989), pp. 1641–1648. DOI: 10.1109/29.46546.

[17] Daniel S. Park et al. "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition". In: *Interspeech 2019* (Sept. 2019). DOI: 10.21437/interspeech.2019-2680. URL: http://dx.doi.org/10.21437/Interspeech.2019-2680.

[18] Sree Hari Krishnan Parthasarathi and Nikko Strom. "Lessons from Building Acoustic Models with a Million Hours of Speech". In: *CoRR* abs/1904.01624 (2019). arXiv: 1904.01624. URL: http://arxiv.org/abs/1904.01624.

[19] Vishal Passricha and Rajesh Kumar Aggarwal. *Convolutional Neural Networks for Raw Speech Recognition*. 2018. DOI: 10.5772/intechopen.80026.

[20] Kihyuk Sohn et al. *FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence*. 2020. arXiv: 2001.07685 [cs.LG].

[21] Chao Sun et al. "A convolutional recurrent neural network with attention framework for speech separation in monaural recordings". In: *Scientific Reports* 11 (1434 2021). DOI: https://doi.org/10.1038/s41598-020-80713-3.

[22] Antti Tarvainen and Harri Valpola. *Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results*. 2018. arXiv: 1703.01780 [cs.NE].

[23] Jakub M. Tomczak and Max Welling. *VAE with a VampPrior*. 2018. arXiv: 1705.07120 [cs.LG].

[24] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

Appendix

We have made the following report edits, as suggested by the peer reviews we have received and our own revisions:

- Rewrote the abstract to clarify it
- Introduction:
  - Ensured that the goals of our project are clearly stated in the introduction
  - Expanded the related work section with more references to other architectures used in speech recognition and the main methods of semi-supervised learning
- Methods
  - Motivated our choice of a small (5%) validation dataset
  - Clarify that we implemented all code from scratch (PyTorch)
  - Further described our hyperparameter tuning
  - Added a brief description of MFCC coefficients and motivated their use
  - Chose a better picture for the LSTM illustration
  - Corrected the reference to SpecAugment and briefly describe it
- Results
  - General cleanup and reordering of the results section so that the best performing architecture is clear
  - Added a visualization of the comparative performance of our architectures
- Conclusion and Discussion
  - Added suggestions for future work
  - Added a comparison of our results to other papers

We found one peer review which gave us 0 points out of 6 for clarity a little unnecessarily harsh, in comparison with the other reviews we received which only found minor faults throughout, nevertheless we did clean up our report quite considerably throughout, removed/combined unnecessary figures and added other. Readers should now find our report with a clear, relevant related works section, a thorough description, visualization and evaluation of results and interesting suggestions for future work we would like to undertake given the opportunity.