

---

# Dual Student Training for Automatic Speech Recognition

---

Andrea Caraffa  
caraffa@kth.se

Kevin Dalla Torre Castillo  
kevindt@kth.se

Simone Porcu  
porcu@kth.se

Franco Ruggeri  
fruggeri@kth.se

## Abstract

Dual Student is a neural network learning algorithm that has recently been a breakthrough for semi-supervised image classification. Two networks are trained on both labeled and unlabeled data, and along with the training the networks share only reliable knowledge to improve each other. This paper employs the experimental method to investigate the effectiveness of Dual Student training for sequential data, such as speech, adapting it for automatic speech recognition. Extensive experiments conducted on the TIMIT dataset show the benefits of this approach to leverage amounts of unlabeled data for frame-based phone classification. Dual Student significantly boosts the performance with respect to training a network using only a small set of labeled utterances. Finally, an improvement is proposed by alternating learning from labeled data and from knowledge exchange. The results show that the novel scheduled learning compared to the standard Dual Student allows achieving better accuracy.

## 1 Introduction

In recent years, several improvements have been done in the field of Automatic Speech Recognition (ASR). The introduction of deep learning techniques has given a new light to the field, enhancing supervised tasks, and revealing the power of semi-supervised models. In the context of phoneme-based classification, both supervised and Semi-Supervised Learning (SSL) can be used. According to nowadays demand and real-world scenarios, data is most commonly unlabeled and a supervised approach only in rare cases can be beneficial. Hence, to exploit data in the best possible way, SSL models are required.

In the context of frame-based phone classification, a SSL solution comes with the work of Salvi et al. [1], who proposed a model based on sparse autoencoders with feed-forward networks and mini-batch stochastic gradient descent, leveraging both labeled and unlabeled data. Other research in the field was conducted with graph-based approaches [2], achieving, to the best of our knowledge, the current state-of-the-art accuracy with the Prior-Regularised Measure Propagation (pMP) algorithm. However, the limitation of this method is its high computational cost.

Another interesting SSL approach employed for image classification is from Ke et al. [3], who aim to improve consistency-based methods, such as VAT [4],  $\Pi$  [5] models and Mean Teacher [6]. Consistency-based methods have achieved state-of-the-art results in SSL. These methods involve two roles, a teacher and a student, and penalize different predictions under small perturbations, according to the *smoothness assumption*. However, since the teacher is actually an exponential moving average (EMA) of the student, the two models are tightly correlated resulting in a performance bottleneck over long training. Dual Student (DS) overcomes this problem by decoupling the two roles, replacing the teacher with another student.

DS's main idea is to share knowledge between two neural networks during the training. Sharing knowledge is not a novel concept and previous work has already been done in this area, mainly for *knowledge distillation*. Caruana et al. [7] have shown a way to efficiently compress knowledge

from a larger model to a smaller one, faster for inference. Hinton et al. [8] further developed this approach using a different compression technique. The effectiveness of knowledge distillation using teacher-student training for building accurate and compact neural networks was also investigated by Fukuda et al. [9]. Nowadays, the student-teacher paradigm has become even more popular for ASR [10, 11, 12, 13]. Sparta et al. [14] applied it using huge amounts of unlabeled data (1 Million hours of speech) and distributed training, reaching impressive results. They trained a very big and slow teacher, unusable in real-life applications due to its size, which was then used to generate labels for the student, smaller and more efficient. This approach was further improved by training the student with probabilistic labels created by the teacher [15]. Unlike these methods, DS has two networks with the same size, the same capability, sharing knowledge with each other.

## 2 Methods

### 2.1 Dual Student training

During DS training two networks improve each other sharing only reliable knowledge, thanks to the concept of *stable sample*, a sample that should satisfy two conditions. First, a small perturbation should not affect the prediction of the sample. Second, the prediction of the sample has to be far from the decision boundary. This means that the sample has a high probability for the predicted label, hence the model is sure about its prediction. It is noteworthy to specify that the notion of *stable sample* applies to the models and not to the DS itself. Therefore, a sample can be stable for one student but may not be for the other. To this regard, the first student shares its knowledge about its *stable samples* to improve the other student. The two aforementioned conditions are encoded in Eq. 1. Given a sample  $x$ , a noisy augmentation  $\bar{x}$ , and  $\theta^i$  the weights of student  $i$ , then the sample  $x$  is stable for student  $i$  according to:

$$\mathcal{R}_x^i = \{\mathcal{P}_x^i = \mathcal{P}_{\bar{x}}^i\}_1 \ \& \ (\{\mathcal{M}_x^i > \xi\}_1 \|\{\mathcal{M}_{\bar{x}}^i > \xi\}_1) \quad (1)$$

where  $\mathcal{P}_x^i$  and  $\mathcal{P}_{\bar{x}}^i$  are the predicted labels of  $x$  and  $\bar{x}$ , respectively, by student  $i$ .  $\mathcal{M}_x^i = \|f(\theta^i, x)\|_\infty$  is the probability of the predicted label,  $\{\text{condition}\}_1$  is a boolean function which returns 1 when the condition is true and 0 otherwise. The hyper-parameter  $\xi$  is a confidence threshold in the range  $[0, 1)$  which defines the minimum probability of the predicted label for the sample to be considered as stable. A lower threshold  $\xi$  implies a softer condition for a *stable sample*.

Whenever a sample is stable for both students, then the Euclidean distance is used to measure the prediction consistency. The stability of sample  $x$  for student  $i$  is defined by:

$$\mathcal{E}_x^i = \|f(\theta^i, x) - f(\theta^i, \bar{x})\|^2 \quad (2)$$

The smaller  $\mathcal{E}_x^i$  is, the more stable sample  $x$  is for student  $i$ . If  $\mathcal{E}_x^i < \mathcal{E}_x^j$ , then student  $i$  will share its knowledge about sample  $x$ , otherwise it is student  $j$  that learns from student  $i$  about sample  $x$ . The distance between the predictions of the two students can be measured using the mean squared error (MSE) as:

$$\mathcal{L}_{mse}(x) = \|f(\theta^i, x) - f(\theta^j, x)\|^2 \quad (3)$$

The stabilization loss for student  $i$  on sample  $x$  is then given by:

$$\mathcal{L}_{sta}^i(x) = \begin{cases} \{\mathcal{E}_x^i > \mathcal{E}_x^j\}_1 \mathcal{L}_{mse}(x), & \text{if } \mathcal{R}_x^i = \mathcal{R}_x^j = 1 \\ \mathcal{R}_x^j \mathcal{L}_{mse}(x), & \text{otherwise} \end{cases} \quad (4)$$

Therefore, if  $x$  is not stable for student  $i$ , then  $x$  contributes in the stabilization loss for student  $i$  only if  $x$  is stable for student  $j$ . On the other hand, if  $x$  is stable for student  $i$ , then sample  $x$  contributes in the stabilization loss for student  $i$  only if  $x$  is even more stable for student  $j$ , that is the prediction of student  $j$  is more reliable. The definition of the stabilization loss ensures that only reliable knowledge is shared between the students and the amount of knowledge shared is controlled by the threshold  $\xi$ .

Besides the stabilization loss, another important role is played by the consistency loss, commonly used in consistency-based methods and defined by:

$$\mathcal{L}_{con}^i(x) = \mathcal{R}(f(\theta^i, x), f(\theta^i, \bar{x})) \quad (5)$$

where  $\mathcal{R}(\cdot, \cdot)$  is a distance measure that can be either MSE or KL divergence. The stabilization constraint drives the students to generalize better given small perturbations on sample  $x$ .

Finally, the overall training loss for student  $i$  can be summarized as follows:

$$\mathcal{L}^i = \mathcal{L}_{cls}^i + \lambda_1 \mathcal{L}_{con}^i + \lambda_2 \mathcal{L}_{sta}^i \quad (6)$$

where  $\mathcal{L}_{cls}^i$  is the standard classification loss while  $\lambda_1$  and  $\lambda_2$  are hyper-parameters to balance the weight of each term. The blueprint of DS training is listed in Algorithm 1.

---

**Algorithm 1:** Training of Dual Student for Semi-Supervised Learning

---

**Require :** Batch  $\mathcal{B}$  containing labeled and unlabeled samples

**Require :** Two independent models  $f(\theta^i)$  and  $f(\theta^j)$

**for each batch  $\mathcal{B}$  do**

    get  $\mathcal{B}_1, \mathcal{B}_2$  from  $\mathcal{B}$  by data augmentation;

**for model in  $\{f(\theta^i), f(\theta^j)\}$  do**

        Calculate  $\mathcal{L}_{cls}$  on labeled samples;

        Calculate  $\mathcal{L}_{con}$  by Eq. 5 between  $\mathcal{B}_1$  and  $\mathcal{B}_2$ ;

**end**

**for each unlabeled sample  $x$  do**

**for model in  $\{f(\theta^i), f(\theta^j)\}$  do**

            Determine whether  $x$  is stable by Eq. 1;

**end**

**if both  $f(\theta^i)$  and  $f(\theta^j)$  are stable for  $x$  then**

            Calculate the stability of  $x$  by Eq. 2;

**end**

        Calculate  $\mathcal{L}_{sta}$  for  $f(\theta^i)$  and  $f(\theta^j)$  by Eq. 4;

**end**

    Update  $f(\theta^i)$  and  $f(\theta^j)$  by the loss in Eq. 6

**end**

---

## 2.2 Data augmentation

A key feature of DS is to augment each batch  $\mathcal{B}$  during the training to get  $\mathcal{B}_1$  and  $\mathcal{B}_2$ . The students, then, aim to predict the same label  $y$  on both the augmented data. Usually, data augmentation for ASR is directly applied to the waveform [16, 17]. Recently, Park et al [18] proposed SpecAugment, a data augmentation method which is instead applied to the filter bank coefficients. We decided to simply apply Gaussian noise over the 39 features, with zero-mean and standard deviation  $\sigma$ , appropriately tuned in the experiments. We are aware that this augmentation method may slightly break the temporal relation encoded in delta and delta-delta, however data augmentation was out of the scope of this project and other augmentation methods can be explored as future work.

## 2.3 Scheduled learning

We improved DS training by introducing a novel idea: scheduled learning. Namely, learning mostly from classification on labeled samples is interleaved with learning from knowledge exchange exploiting unlabeled samples. The reason that drives this variation is that, once a student has learnt from the other student, it may be better to exploit this new acquired knowledge focusing more on the labeled samples. Lately, the model can exchange further (and better) knowledge with the other student and so on. The modification was implemented updating the values of  $\lambda_1$  and  $\lambda_2$  at each epoch according to a cyclical schedule, while in [3] they follow a ramp-up only in the first 5 epochs and then they remain constant.

We proposed two new schedules: triangular cycling and sinusoidal cycling, both shown in Figure 1 and compared to the original ramp-up. The values on the y axis, in the range  $[0, 1]$ , must be multiplied by two scale factors,  $\lambda_1^{max}$  and  $\lambda_2^{max}$ , to get respectively the correct  $\lambda_1$  and  $\lambda_2$  along the training. When  $\lambda_1$  and  $\lambda_2$  reach their minimum values, the students are optimized by minimizing mainly the classification loss. On the other hand, when  $\lambda_1$  and  $\lambda_2$  reach their peak, knowledge exchange comes into play. In the first period every schedule starts from 0, since it is useless to share knowledge at the very beginning when both students still have not learnt anything. Contrarily, for the remaining periods, the cyclical schedules have their minimum value set to 0.5 instead of 0 to avoid focusing exclusively on the labeled data, which would lead to overfitting, decreasing the overall performance.

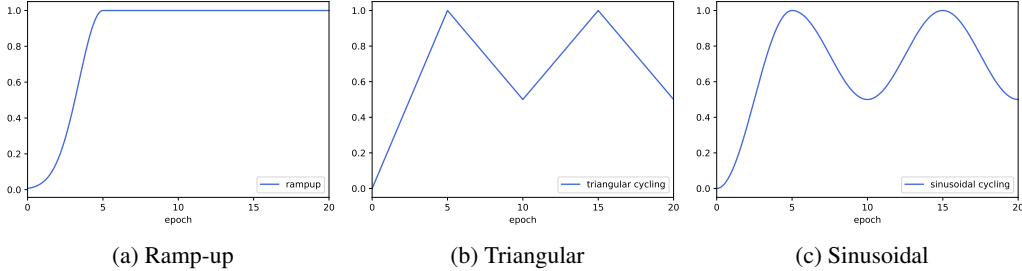


Figure 1: Learning schedules to control the values of  $\lambda_1$  and  $\lambda_2$ . Triangular and sinusoidal cycling are the novel learning schedules proposed by us.

### 3 Experiments

#### 3.1 Data

We performed our experiments on the TIMIT dataset [19], which contains utterances from native English speakers along with phonetic transcriptions. Following standard recipes [20, 21] and previous work [1], we excluded glottal stop segments and SA sentences (read by all the 630 speakers), and we used the core test set, consisting of 192 sentences. For tuning hyper-parameters and monitoring the training, we drew out from the training set 184 sentences as validation set. In order to achieve a less biased generalization assessment in the validation phase, this split was done guaranteeing that no speaker appears in both sets. The inputs to the networks are 39-dimensional feature vectors, each of which contains 13 MFCCs, delta and delta-delta features. The feature extraction was performed with a Hamming window of length 30 ms and shift of 10 ms, resulting in a total of 1049763 training frames, 54912 validation frames and 56623 test frames. For training we used the standard set of 48 phones, while in the evaluation phase we mapped them to 39 phones, as done in [1, 20, 15]. To experiment the benefits of unlabeled data, we removed the targets to a percentage of frames. We varied the percentage of labeled samples to 1%, 3%, 5%, 10%, 20% and 30%. Importantly, each utterance can only be either fully labeled or fully unlabeled. The whole procedure and the resulting numbers are in line with previous work [1, 15] and so a fair comparison can be done.

The features were then normalized. Three different types of normalization were considered: over the whole training set, over each speaker separately, and each utterance individually. Some preliminary experiments on a 3-layer mono directional Long Short Term Memory (LSTM) showed that speaker-dependent normalization significantly achieves better results, hence, we adopted this normalization for the rest of the experiments.

#### 3.2 Architectures

In the experiments we considered two architectures, both consisting of 3 LSTM layers of 96 units followed by a fully connected layer. The only difference between the two used architectures is the type of LSTM employed. Indeed, one consists of mono directional LSTM layers whereas the other consists of bidirectional LSTM layers. The number of layers and number of units were chosen for a fair comparison with the work by Lin et al. [15].

We tested all the combinations of the architectures for the two students. Therefore, one combination uses mono directional LSTM layers in both students (mono-DS), while the other uses bidirectional LSTM layers (bi-DS). Finally, a third combination, called Imbalanced Student (IS), has one of each architecture.

Moreover, to analyze the benefits of unlabeled data, we set as baselines the single architectures with mono directional LSTM layers (mono-LSTM) and bidirectional LSTM layers (bi-LSTM), trained in a fully-supervised way.

#### 3.3 Hyper-parameters

In line with the work by Lin et al. [15] the batch size was set to 100. The batch contains both labeled and unlabeled data proportionally to the composition of the training set. Consequently,

Table 1: Network architecture, where L represents the length (number of frames) of an utterance.

Layer (type)	Output Shape
Utterance $\mathbf{x}$	$L \times 39$
LSTM	$L \times 96$
LSTM	$L \times 96$
LSTM	$L \times 96$
Dense	$L \times 48$

Table 2: Optimal hyper-parameters found for mono-DS with percentage of labeled data equal to 30% on the TIMIT dataset.

schedule	$\sigma$	$\xi$	$\lambda_1^{max}$	$\lambda_2^{max}$	con. loss	val. accuracy (%)
ramp-up	0.5	0.3	10	100	MSE	73.32
triangular	0.3	0.3	10	100	MSE	72.83
sinusoidal	0.5	0.4	10	100	MSE	73.60

in mono-LSTM and bi-LSTM, which contain only labeled data, the number of samples actually considered in each batch was equal to  $\%labeled \times 100$ . In this way, the gradient updates for labeled samples were the same for both DS and supervised training.

We adopted Adam with weight decay, a well-known optimizer which performs better than the simple stochastic gradient descent. We selected learning rate  $\eta = 0.01$  and weight decay  $wd = 10^{-4}$  with preliminary experiments on mono-LSTM. For all the other hyper-parameters, default values were used. Finding optimal settings for the optimizer was out of scope, since our goal is not to provide state-of-the-art SSL results on TIMIT, but rather to show the benefits of DS training for sequential data.

DS algorithm introduces four additional hyper-parameters not used in standard supervised learning procedure:  $\lambda_1^{max}$ ,  $\lambda_2^{max}$ ,  $\sigma$ , and  $\xi$ . We performed an exhaustive grid search to tune them as well as selecting between MSE and KL as consistency loss for each of the three learning schedules. The grid search was only done for mono-DS with 30% of labeled data. Given the amount of hyper-parameters, tuning them also for bi-DS, IS and different percentages of labeled data was unfeasible in the time-scope of the project.

In the grid search the training was run over 20 epochs.  $\lambda_1^{max}$  and  $\lambda_2^{max}$  were searched in log space, whereas  $\sigma$  and  $\xi$  in linear space. A summary of all the configurations tested in the grid search is shown in Figure 2. In total, over 1500 hyper-parameter settings were considered. It is important to mention that the most-right plots reveal the correlation between the two students, where it can be easily seen that for most of the cases each student learns as much as the other. The top 10 hyper-parameter settings are shown in Figure 3, and only triangular and sinusoidal cyclical schedules fall into this ranking. The 10 most promising configurations for each learning schedule were further tested over a longer training of 100 epochs and the final optimal hyper-parameters are shown in Table 2.

### 3.4 Practical setup

The work was implemented from scratch using TensorFlow, a powerful and open-source deep learning framework, and is accessible on GitHub<sup>1</sup>. The experiments were run on only one GPU, using Google Colab and Google Cloud Platform.

## 4 Results

The optimal hyper-parameters previously found were used to train the models with different percentages of labeled data. Table 3 shows a comparison between the baseline mono-LSTM and mono-DS.

<sup>1</sup><https://github.com/simone-porcu/DT2119-Final-Project>

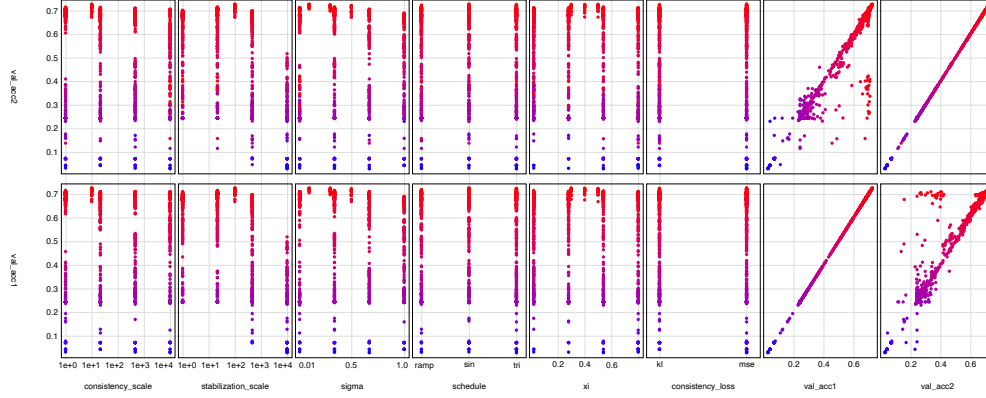


Figure 2: Scatter plot of more than 1.500 configurations tested in the grid search.

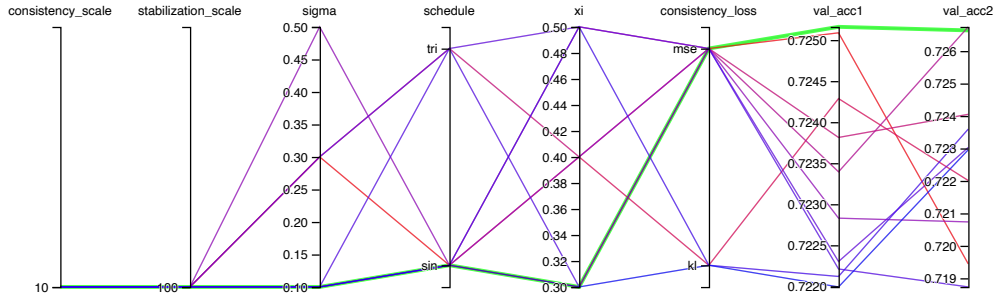


Figure 3: Parallel coordinate plot showing the top 10 configurations from the grid search, the one with the highest validation accuracy is highlighted in green.

The comparison also takes into consideration the three different schedules investigated. Mono-DS with triangular cyclical schedule is the best model in the less supervised experiments, i.e. with 1%, 3%, 5% and 10% of labeled data, gaining  $\approx 9\%$  test accuracy against mono-LSTM. On the other hand, for 20% and 30% of labeled data, mono-DS with ramp-up schedule is the best one, gaining 7.96% and 6.36%, respectively, over the mono-LSTM.

Table 4 compares the baseline mono-LSTM and IS. It is noteworthy to highlight that the results for IS refer to the mono directional student. Indeed, the goal of the imbalanced configuration is to train an efficient small student (mono directional) by letting it learn together with a bigger one (bidirectional). The conclusions are similar to the mono-DS, indeed apart from 30% of labeled data, where IS with sinusoidal schedule achieved the highest accuracy, IS with the triangular schedule outperforms all the other models. Moreover, we can observe that even though the bidirectional student should pump up the mono directional performance in the IS set up, this is not the case, indeed there is not such a great difference between the accuracy achieved by mono-DS and IS.

Furthermore, additional experiments compared bi-LSTM to bi-DS, as shown in Table 5. Also in this case bi-DS outperforms bi-LSTM for all percentages of labeled samples. In addition, as expected, bi-DS is better than mono-DS in terms of accuracy, given the relative capacity, at the cost of a less scalable solution.

The results also point out that for small percentages of labeled data, i.e 1%, 3% and 5%, mono-DS outperforms bi-LSTM, emphasizing how a smaller network can beat a more powerful network thanks to DS training, since mono-DS with triangular cyclical schedule (see Table 3) achieved even better results than bi-LSTM (see Table 5).

Finally, mono-DS and IS were compared to SSL renowned methods tested on the TIMIT dataset, as shown in Table 6. We can notice that both mono-DS and IS achieved better results than previous work [1, 15], including pMP [2].

Table 3: Results on frame-based phone classification on the test and validation sets on the TIMIT dataset for mono-LSTM and mono-DS, for each learning schedule.

Labeled	mono-LSTM		mono-DS ramp-up		mono-DS tri		mono-DS sin	
Obs. (%)	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.
1	50.81	51.59	56.34	56.20	56.66	<b>56.69</b>	55.77	55.63
3	57.13	56.73	62.27	61.99	63.14	<b>63.08</b>	62.59	61.93
5	59.74	59.65	65.00	64.27	65.34	<b>65.22</b>	64.68	64.62
10	62.84	61.86	68.15	67.07	68.33	<b>67.82</b>	67.81	67.29
20	66.83	65.39	71.39	<b>70.60</b>	71.03	70.12	70.91	70.46
30	69.21	67.91	73.32	<b>72.23</b>	72.83	71.62	73.60	72.20

Table 4: Results on frame-based phone classification on the test and validation sets on the TIMIT dataset for mono-LSTM and IS (the accuracy of the mono directional student is shown), for each learning schedule.

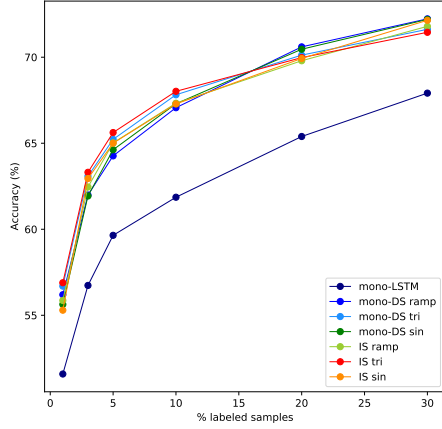
Labeled	mono-LSTM		IS ramp-up		IS tri		IS sin	
Obs. (%)	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.
1	50.81	51.59	56.59	55.86	56.78	<b>56.89</b>	56.15	55.30
3	57.13	56.73	63.05	62.46	63.97	<b>63.31</b>	63.32	62.95
5	59.74	59.65	65.56	65.02	66.22	<b>65.62</b>	65.11	65.01
10	62.84	61.86	68.25	67.33	68.76	<b>68.02</b>	67.97	67.28
20	66.83	65.39	71.52	69.69	71.56	<b>69.98</b>	71.63	69.93
30	69.21	67.91	73.39	71.79	72.99	71.45	73.22	<b>72.15</b>

Table 5: Results on frame-based phone classification on the test and validation sets on the TIMIT dataset for bi-LSTM and bi-DS, for each learning schedule.

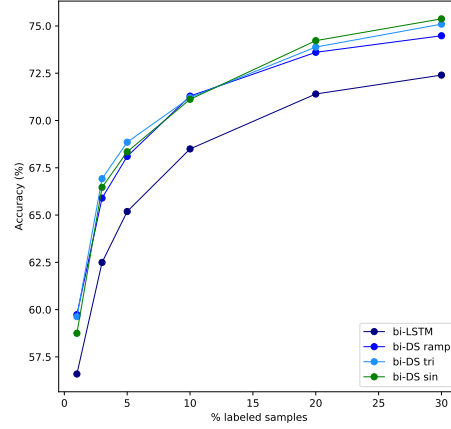
Labeled	bi-LSTM		bi-DS ramp-up		bi-DS tri		bi-DS sin	
Obs. (%)	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.	val. acc.	test acc.
1	56.48	56.60	59.70	<b>59.74</b>	59.65	59.63	58.80	58.75
3	63.99	62.50	67.16	65.90	67.29	<b>66.92</b>	67.14	66.47
5	66.28	65.19	69.22	68.10	69.28	<b>68.85</b>	69.34	68.35
10	69.40	68.50	72.21	<b>71.29</b>	72.40	71.21	72.77	71.12
20	72.97	71.40	75.39	73.61	74.72	73.88	75.37	<b>74.22</b>
30	73.57	72.40	75.90	74.48	76.21	75.09	76.74	<b>75.38</b>

Table 6: Accuracy rates (%) for frame-based phone classification on the TIMIT dataset for mono-LSTM, SSSAE [1], pMP [2], Teacher-Student [15], mono-DS and IS. The optimal learning schedule was chosen for each percentage of labeled samples according to the values in Table 3.

Method	Reference	10% labeled	30% labeled
Test accuracy (%)			
mono-LSTM	this work	61.86	67.91
SSSAE	[1]	67.03	69.65
pMP	[2]	67.22	71.06
Teacher-Student	[15]	67.15	70.78
mono-DS	this work	67.82	<b>72.23</b>
IS	this work	<b>68.02</b>	72.15



(a) Mono-LSTM, mono-DS and IS.



(b) Bi-LSTM and bi-DS.

Figure 4: Frame-based phone recognition accuracy (%) for different percentages of labeled samples on the TIMIT database.

## 5 Conclusions

A great number of experiments was conducted for different hyper-parameters and architectures. The results were consistent through all the experiments for different percentages of labeled data. DS always outperforms the standard fully-supervised learning procedure given the same architecture and hyper-parameters. Therefore, we can conclude that DS approach is beneficial to exploit large amounts of unlabeled samples even for handling sequential data such as speech. We also introduced a novel scheduled learning, employing triangular and sinusoidal cycling, which overall showed slightly better results with respect to ramp-up.

Further experiments could be to fine-tune the hyper-parameters for each percentage of labeled data, taking in consideration also the learning rate as well as different optimizers. Future work may also focus on other data augmentation methods. Finally, experiments on the period of the sinusoidal/triangular cycle may investigate its influence on the performance, whose values were instead fixed in this project.



## References

- [1] Akash Kumar Dhaka and Giampiero Salvi. “Sparse Autoencoder Based Semi-Supervised Learning for Phone Classification with Limited Annotations”. In: (2017).
- [2] Y. Liu and K. Kirchhoff. “Graph-based semi-supervised learning for phone and segment classification”. In: (Jan. 2013), pp. 1840–1843.
- [3] Zhanghan Ke et al. *Dual Student: Breaking the Limits of the Teacher in Semi-supervised Learning*. 2019. arXiv: 1909.01804 [cs.LG].
- [4] Takeru Miyato et al. *Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning*. 2018. arXiv: 1704.03976 [stat.ML].
- [5] Samuli Laine and Timo Aila. *Temporal Ensembling for Semi-Supervised Learning*. 2017. arXiv: 1610.02242 [cs.NE].
- [6] 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].
- [7] Cristian Buciluundefined, Rich Caruana, and Alexandru Niculescu-Mizil. “Model Compression”. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’06. Philadelphia, PA, USA: Association for Computing Machinery, 2006, pp. 535–541. ISBN: 1595933395. DOI: 10.1145/1150402.1150464. URL: <https://doi.org/10.1145/1150402.1150464>.
- [8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML].
- [9] T. Fukuda et al. “Efficient Knowledge Distillation from an Ensemble of Teachers”. In: *INTER-SPEECH*. 2017.
- [10] Lei Jimmy Ba and Rich Caruana. *Do Deep Nets Really Need to be Deep?* 2014. arXiv: 1312.6184 [cs.LG].
- [11] Jinyu Li et al. *Large-Scale Domain Adaptation via Teacher-Student Learning*. 2017. arXiv: 1708.05466 [cs.CL].
- [12] Liang Lu, Michelle Guo, and Steve Renals. *Knowledge Distillation for Small-footprint Highway Networks*. 2016. arXiv: 1608.00892 [cs.CL].
- [13] G. Tucker et al. “Model Compression Applied to Small-Footprint Keyword Spotting”. In: *INTERSPEECH*. 2016.
- [14] Sree Hari Krishnan Parthasarathi and Nikko Strom. *Lessons from Building Acoustic Models with a Million Hours of Speech*. 2019. arXiv: 1904.01624 [cs.LG].
- [15] Chun Hung Lin et al. *Semi-supervised learning with soften probabilistic labels*. <https://github.com/chris4540/DT2119-Final-Project>. 2019.
- [16] Michael F O’Rourke and Alfredo L Pauca. “Augmentation of the aortic and central arterial pressure waveform”. In: *Blood pressure monitoring* 9.4 (Aug. 2004), pp. 179–185. ISSN: 1359-5237. DOI: 10.1097/00126097-200408000-00002. URL: <https://doi.org/10.1097/00126097-200408000-00002>.
- [17] Navdeep Jaitly and E. Hinton. “Vocal Tract Length Perturbation (VTLP) improves speech recognition”. In: 2013.
- [18] 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>.
- [19] John S. Garofolo et al. “DARPA TIMIT: acoustic-phonetic continuous speech corpus CD-ROM, NIST speech disc 1-1.1”. In: 1993.
- [20] D. Povey et al. “The Kaldi Speech Recognition Toolkit”. In: 2011.
- [21] Yajie Miao. *Kaldi+PDNN: Building DNN-based ASR Systems with Kaldi and PDNN*. 2014. arXiv: 1401.6984 [cs.LG].