I. Introduction to the TIMIT dataset

For pre-processing, each TIMIT utterance of waveforms is converted into acoustic features. The selected form is Log Mel filter banks (FBank), which projects the spectrum onto a Mel-scaled filter bank with a logarithmic transform. The original TIMIT corpus was mapped to a smaller 39 phone set, plus a silence phone "sil." The distribution of the phonemes in the training set is heavily imbalanced, as seen in Figure 1. Phones such as "ih" and "ah" dominate while phones such as "uh" and "oy" are sparse. This imbalance can affect the model's performance, leading to potential phoneme confusions for the underrepresented phones.

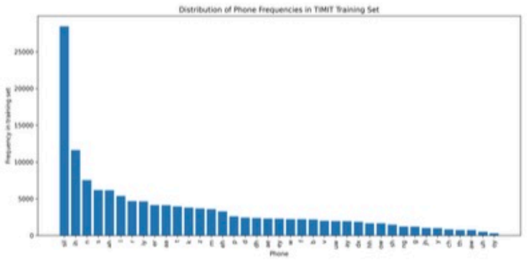

Fig. 1. Distribution of phone frequencies in the training set

II. Training

A. Training Baseline Model

The baseline model is a single layer 128-dim BiLSTM and was trained using stochastic gradient descent (SGD) on the 23-dimensional FBank features. The performance metrics are listed in Table I. The model achieved a test PER of 30.12% and correctly recognizes 72.62% of phonemes in the test set. The validation PER was very close, at 29.13%. This means the model had good generalization across unseen speakers and had not overfitted, as seen in Fig. 2. One limitation of this model is its high substitution rate over insertion, meaning it faced some amount of confusion between acoustically similar phones. This, alongside the moderate PER, can be improved using regularization and further optimization.

B. Regularization

1) Dropout: The first regularization method tested was the dropout method. Dropout was applied to the outputs of the BiLSTM before the projection layer, acting as feed-forward regularization. Various dropout probabilities p were tested, as listed in table II. As the dropout probability increases, the training loss rises while the validation loss has little variation. The dropout method is therefore affecting how well the model fits the training data but having a little-to-no benefit in improving validation loss on TIMIT.

Metric	**Value**
Training Loss	0.74
Validation Loss	0.966
Validation PER	29.13%
Test PER	30.12%
Test Error Breakdown	
Substitution Rate	17.12%
Deletion Rate	10.25%
Insertion Rate	2.74%
Correct Phones (COR)	72.62%

TABLE I

Baseline end-to-end CTC model performance on TIMIT using 23-dim FBank features and 1-layer BiLSTM.

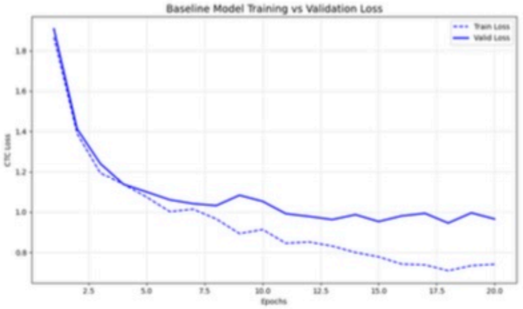

Fig. 2. Baseline Model Training/Validation Loss

Introducing small to moderate dropout (0.05, 0.10) helps in reducing the training and validation loss gap, which improves regularization and leads to the best test PER (29.65%), an improvement from the baseline. A higher dropout (0.30) increases both the training and validation loss, a symptom of underfitting. This is because a high dropout increases the models uncertainty of phoneme alignment. When certain features are zeroed out at each time step, the model can struggle to predict a strong sequence of phonemes. This forces the model toward the blank token that leads to an increase in loss. The moderate dropout provides the best balance, as it benefits the model without hurting the learning of alignments.

2) Gradient Clipping: Another regularization method is gradient clipping. Various maximum gradient norms were tested, as listed in table III. The current best model utilized a dropout probability of 0.1, which is kept for these runs. Smaller maximum gradient norms leads to stronger gradient clipping. As the maximum gradient norm decreases, the generalization of the model improves and leads to lower

Dropout p	Tr. Loss	Val. Loss	Val. PER	Test PER
0.0 (baseline)	0.742	0.967	29.13	30.12
0.05	0.828	0.979	30.16	31.14
0.10	0.846	0.968	29.58	**29.65**
0.30	0.933	0.975	30.86	31.80

TABLE II

DROPOUT PERFORMANCE ON BILSTM.

loss and PER for both training and testing. This reduction in training loss indicates that clipping prevents high gradient spikes, and instead having more controlled gradient updates.

Max-Norm	Tr. Loss	Val. PER	Test PER
No clipping	0.84590	29.58	29.65
5	0.83263	29.55	29.51
2	0.73983	28.58	29.34
1	0.74403	27.76	**29.03**

TABLE III

EFFECT OF GRADIENT CLIPPING (WITH DROPOUT = 0.1).

The best model thus far achieves a test PER of 29.03%, with a max-norm of 1 and dropout probability of 0.1.

C. Optimizer

1) SGD vs Adam: Using a dropout probability of 0.1, two optimizers were tested on the model: Stochastic gradient descent (SGD) and adaptive moment estimation (Adam). Various learning rates were tested independently for both optimizers, as seen in table IV.

Adam is very sensitive to the set learning rate in CTC training. With low learning rates (0.0001, 0.0005), the model has poor performance. The losses are over 1.35, and the test PER is high (approx. 52%), showing the model is performing only slightly better than random guessing on phoneme alignments. This is likely because of the blank token dominating gradient updates and Adam's per-parameter learning rate adaption amplifying this. With a high learning rate of 0.001, Adam helps the model overcome this and achieve the best performance among the optimizer experiments: a test PER of 29.57%.

Optim.	LR	Tr. Loss	Val. Loss	Val. PER	Test PER
Adam	0.0001	1.35016	1.36861	52.24	52.15
Adam	0.0005	1.35012	1.36862	52.23	52.17
Adam	0.0010	**0.74070**	**0.94405**	**28.31**	**29.57**
SGD	0.0500	1.09162	1.09997	35.03	35.95
SGD	0.1000	0.92611	0.99145	31.04	32.18
SGD	0.5000	**0.89383**	**0.99274**	**30.48**	**30.10**

TABLE IV

COMPARISON OF SGD AND ADAM WITH DIFFERENT LEARNING RATES (DROPOUT = 0.1).

SGD has more stable behavior across learning rates, since it applies the same update to all parameters. As the learning rate increases from 0.05 to 0.5, the training/validation losses and validation/test PER show a steady decrease and increase, respectively. This gradual improvement shows better alignment learning, as seen by the mean performance of SGD vs Adam in Fig. 3.

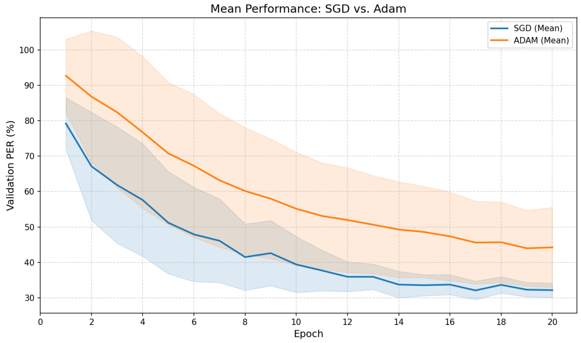

Fig. 3. Mean Performance for SGD and Adam (with a shaded area for variance.)

2) Learning Rate Scheduler: A learning rate scheduler is used during SGD training, where the learning rate is halved whenever validation loss increases. Using a dropout probability of 0.1, the performance is recorded in table V.

In prior runs, validation PER is moderate but begins to plateau in the middle of training, which indicates diminishing returns from large update steps. In this learning rate scheduler run, each learning rate drop shows a decrease in validation loss and PER, as seen in Fig. 4. This proves that a constant learning rate can become suboptimal if the model approaches a local minimum. With the final test PER of 28.20% and a close validation PER, the learning rate scheduler provides strong generalization and an improvement in performance thus far. This is because the scheduler allows SGD to use large learning rates early to gain quick progress, then switch to smaller learning rates for fine-tuning once the progress halts. This is beneficial for CTC training, where more stable alignment updates are required in later stages to sharpen the probabilities.

Epoch of Drop	LR Pre	LR Post	Val. Loss	Val. PER
12	0.50	0.25	1.017	31.92
16	0.25	0.125	0.918	28.90
20	0.125	0.0625	0.895	27.63
Final Test PER (%)				**28.20**

TABLE V

LEARNING RATE REDUCTIONS AND EFFECT ON PERFORMANCE (SGD, DROPOUT = 0.1).

D. Model Complexity

1) 2 Layers and Wider Layers: Two BiLSTM architectures are compared: a single layer BiLSTM with 167k parameters, and a 2-layer BiLSTM with 562k parameters. Dropout is fixed at 0.1, and an SGD learning rate scheduler is used. As seen in table VI, increasing the layers of the model leads to better performance, with the model achieving a best-yet test PER of 26.81%. The training loss for the 2-layer model is higher than that of the 1-layer model, but it achieves a lower

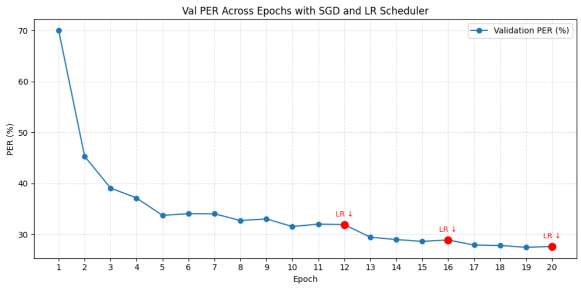

Fig. 4. Val PER with SGD and LR Scheduler

validation loss and a smaller training-validation loss gap, as seen in Fig. 5. This means that the deeper architecture allows a better representation of speech features that generalizes better to unseen data. This 1.4% PER improvement from adding a second layer exceeds the gains from regularization alone (dropout and scheduler combined), meaning that the model's capacity was the main performance bottleneck for the 1-layer architecture.

Model	Params	Tr. Loss	Val. Loss	Val. PER	Test PER
1-layer	166952	0.69734	0.89547	27.63	28.20
2-layer	562216	0.72258	0.85435	**26.28**	**26.81**

TABLE VI
COMPARISON OF 1-LAYER VS. 2-LAYER BILSTM MODELS (DROPOUT = 0.1, SGD WITH SCHEDULER).

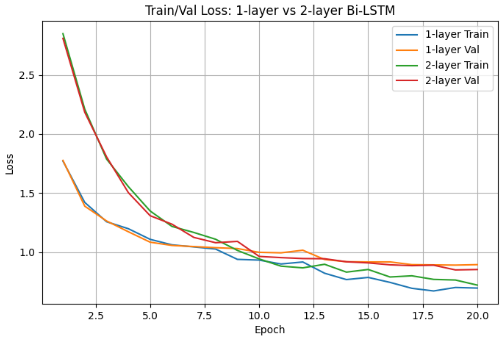

Fig. 5. Train/Val Loss for Single vs Double Layer BiLSTM

The next architecture adjustment made was layer width. The layer width was increased from 128 to 512 for both the single and 2-layer models. This largely increases the parameter count (2.2M-8.5M parameters). The 1-layer LSTM performance worsens with a wide layer compared to the narrow model (28.55% vs 28.20%). The 2-layer wide (512) model achieves the best PER of 24.50% but at nearly 15 times the parameters of the 2-layer narrow model. Added width is therefore most effective when combined with added depth, rather than width alone. With 2 layers, the first layer can learn the main acoustic patterns, and the second layer can model phonetic transitions. A single layer must handle both simultaneously, so adding width offers capacity the model cannot use in an efficient way. The Pareto frontier in Fig. 6 shows that the 2-layer narrow and 2-layer wide configurations represent the optimal trade-off between capacity and accuracy, while the 1-layer wide model falls significantly above this frontier.

Model	Params	Tr. Loss	Val. Loss	Val. PER	Test PER
1-layer wide (512)	2240552	0.690	0.907	27.68	28.55
2-layer wide (512)	8540200	0.500	0.793	**23.58**	**24.50**
2-layer normal (128)	562216	0.723	0.854	26.28	26.81

TABLE VII
COMPARISON OF WIDE AND NORMAL ARCHITECTURES.

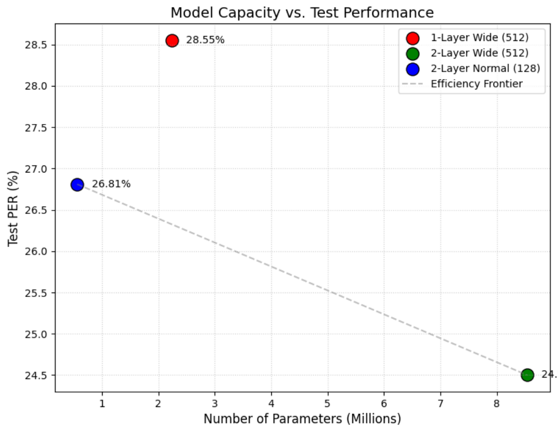

Fig. 6. Pareto Frontier for Wide vs Narrow Models

2) Uni-Directional LSTM: Another architecture adjustment tested was the use of uni-direction as opposed to a bi-directional LSTM. This is to simulate streaming ASR constraints, where future context is unavailable. As seen in table VIII, the uni-directional models have much higher PER than their bidirectional counterparts. The 1-layer uni-directional model has the worst performance, with a test PER of 36.13%. Increasing the depth to 2 layers improves performance, with a test PER of 31.02%, but the uni-directional models still lag behind the BiLSTMs that achieved a test PER of 24.50%. This 6-7% PER gap between uni/bi-directional models reflects the importance of future phonetic context in English. Phonemes are classified differently depending on following information (such as vowels ”ih” and ”eh”), and without future frames, the UniLSTM cannot distinguish these variations well.

The error rates for both models, seen in Fig. 7, proves the UniLSTM's struggle. It has higher substitution and deletion rates compared to the BiLSTM. The increased substitution errors shows it frequently confused one sound for another. The higher deletion rate reflects less stable CTC alignments, since the UniLSTM is more likely to emit blank symbols prematurely.

One observation is that adding a second layer benefits the UniLSTM more than the BiLSTM (5.1% vs 1.4% improvement). This can mean that the second layer is trying to compensate for the lack of future information. The BiLSTM already has access to that, so the additional layer is less beneficial.

Model	Params	Tr. Loss	Val. Loss	Val. PER	Test PER
2-layer uni.	215592	0.886	0.968	30.90	31.02
1-layer uni.	83496	1.050	1.103	35.07	36.13

TABLE VIII
PERFORMANCE OF UNIDIRECTIONAL LSTM MODELS.

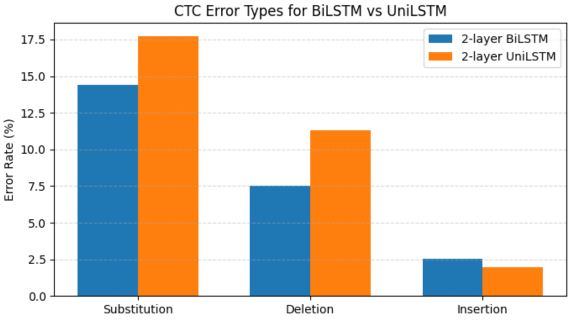

Fig. 7. CTC Error Types for BiLSTM vs UniLSTM

E. Data Augmentation

Data augmentation was performed in the form of speed perturbation to improve performance. The original training data of speed 1.0x was joined by sped up (1.1x) and sped down (0.9x) versions of the audio. This new training data was used on both the wide and narrow 2-layer BiLSTMs, with the results recorded in table IX.

The models trained on the augmented data outperformed the same architectures trained on the original data. Speed perturbation at 0.9x and 1.1x simulates the natural variation in the rate of speaking, and gives the model triple the amount of training data. The smaller 128-dim model has the largest improvement (2.11% reduction in PER), meaning the smaller models were previously limited by their width in capturing speaker differences. For the larger models, the gains are smaller likely due to diminishing returns and an already strong baseline performance. As seen in Fig. 8, data augmentation significantly reduces training and validation loss. It causes a larger training-validation loss gap due to the validation data being unperturbed, yet still is lower than the losses for the model trained on the normal data.

Model	Aug.	Tr. Loss	Val. PER	Test PER
2-Layer Norm (128)	No	0.7226	26.28	26.81
	Yes	0.3790	23.80	**24.70**
2-Layer Wide (512)	No	0.4998	23.58	24.50
	Yes	0.1495	22.98	**24.08**

TABLE IX
IMPACT OF SPEED PERTURBATION DATA AUGMENTATION ON LSTM ARCHITECTURES

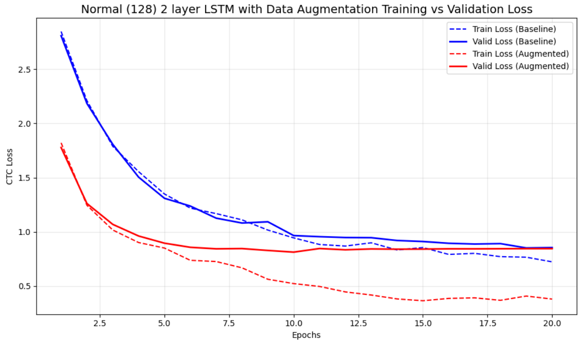

Fig. 8. Normal (128) 2 layer LSTM with Data Augmentation Training vs Validation Loss

III. DECODING

A. Visualization

To visualize the output probability distributions seen in Fig. 9, a single validation utterance (FAKS0_SI943.WAV) was selected, and a forward pass was run to obtain the frame-level posterior probabilities over the vocabulary.

The heatmap shows sharp probability peaks for continuant sounds, such as ”w,” ”l,” and ”z”. These sounds are usually held longer which means the model has more frames to classify them. The blank symbol dominates a majority of the frame probabilities. This is expected CTC behavior, since the model assigns high confidence to blanks by default, and concentrates probability only when each phoneme is most likely to occur. For each frame at a time, only one or two phonemes have high probability (besides blank). This shows the model is confident about predicting when each phoneme is occurring.

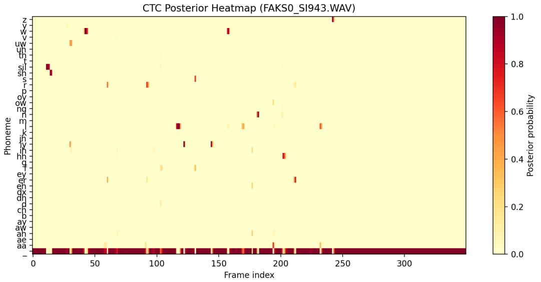

Fig. 9. CTC Heatmap

Another form of visualization is a spectrogram. Fig. 10 shows a log Mel filterbank (FBank) spectrogram of a single utterance with phoneme boundaries from Viterbi decoding under CTC. The background heatmap’s color intensity corresponds to log-scaled acoustic energy in each time frequency bin, and the vertical lines and labels indicate where the model believes each phoneme begins based on the most likely CTC alignment path.

The fricatives (”s”, ”z”, ”sh”) appear in brighter regions because their high frequency noise is stronger and more concentrated in fewer frequency bins, which makes them easier anchor points for CTC. The vowels (”iy”, ”aa”, ”eh”) are smoother and spread-out across many frequency bands, since they use an open vocal tract.

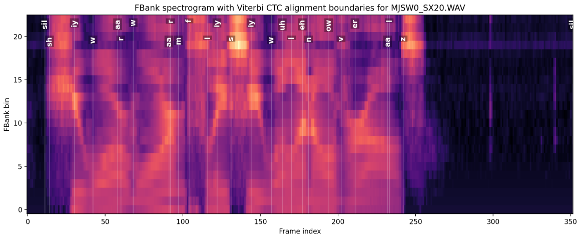

Fig. 10. FBank spectogram with Viterbi CTC Alignment Boundaries

B. Blank Penalty

To address the large amount of blank outputs, different blank penalties are tested that subtract from the blank posterior probabilities. Increasing the blank penalty consistently reduced deletion errors, as seen in table X. As the deletion rate decreases, insertion rate increases, indicating a trade-off where suppressing blanks encourages more frequent symbol outputs. Overall PER improved for moderate penalties but worsened at higher ones, where insertion errors dominated. A small blank penalty, such as 0.5, provides a better balance between deletions and insertions and improves decoding accuracy (PER of 24.30%). An excessive penalization of blanks can disrupt the CTC alignments.

A high blank penalty can improve the performance of the UniLSTM, which suffered from a high deletion rate as seen in Figure 7.

Blank penalty	SUB	DEL	INS	COR	PER
0.00	14.42	7.53	2.54	78.05	24.49
0.50	14.80	6.09	3.41	79.11	24.30
1.00	15.05	4.88	4.60	80.07	24.53
1.50	15.24	3.92	5.97	80.85	25.13
2.00	15.31	3.11	7.77	81.59	26.18

TABLE X
EFFECT OF BLANK PENALTY ON CTC DECODING PERFORMANCE

C. Confusion Matrix

The phoneme-level confusion matrix in Fig. 11 was created using edit-distance backtracking to align reference and predicted phoneme sequences. Most of the confusions are from predictable acoustic patterns. The model is most often confused between vowels, such as “ah” with “ih” and “eh” with “ih,” as they sound the most similar. It also confuses fricatives, such as “s” with “z,” since their differing voicing can be hard to detect in noise. These confusions show that the model is able to learn between phonetic classes but struggles to distinguish within them.

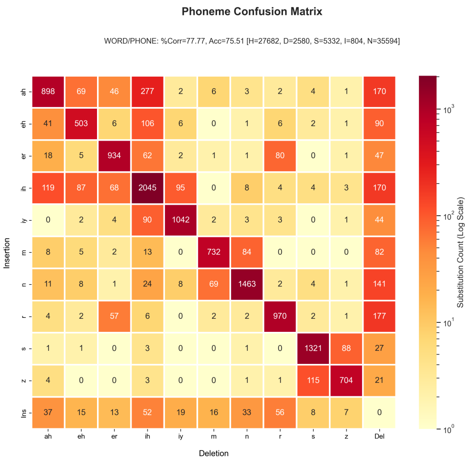

Fig. 11. Phoneme Confusion Matrix

IV. Best Performing Model (and hour allocation strategy)

A. Best Performing Model

The best model system found during these experiments is the 2-layer 128-dim BiLSTM using 23-dim FBANK features, trained on speed perturbed data and a dropout probability of 0.1. This model uses 562k parameters and achieves a test PER of 24.70%. This is right along the average for published error rates on TIMIT, which have achieved PERs from 25.17% to 24.4% [1]. As seen in Fig. 12, the model reaches convergence around epoch 14 while the baseline continues to fluctuate. Another improvement compared to the initial model is that deletion rate dropped from 10.25% (baseline) to 6.84%, as seen in Figure 13. This improvement can be traced back to the increase of layers and data augmentation. Adding a layer reduced deletions to 8.8%, and training data augmentation reduced it to 6.84%. This shows that the 1-layer model lacked capacity to emit phones and defaulted to blanks, and that augmentation taught the model more with the speaking rate variation that it did not have before.

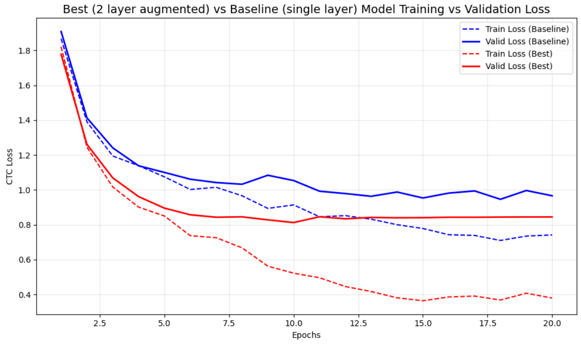

Fig. 12. Best vs Baseline Model Training/Validation Loss

While the 2-layer wide (512) model trained on the speed perturbed data achieves the lowest PER overall, its computational cost and parameter count of over 8 million are disproportionate to the gain.

Metric	**Value**
Training Loss	0.379
Validation Loss	0.845
Validation PER	23.80%
Test PER	24.70%
Test Error Breakdown	
Substitution Rate	14.95%
Deletion Rate	6.84%
Insertion Rate	2.91%
Correct Phones (COR)	78.21%

TABLE XI

BEST END-TO-END CTC MODEL ON TIMIT

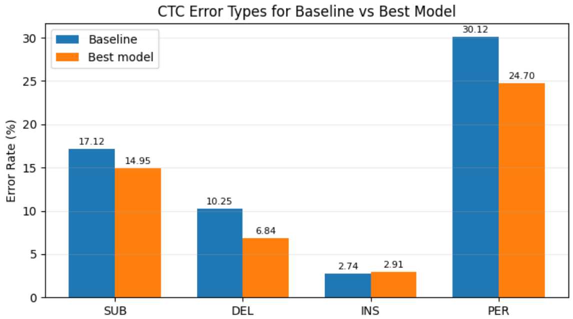

Fig. 13. CTC Error Types for Baseline vs Best Model

B. Hour Allocation

12 GPU hours were used in total. Approximately 4 hours were allocated for regularization and optimization experiments (including dropout and learning-rate scheduling), 4 hours for model complexity (depth and width) and data augmentation, and 2 hours for decoding and visualization. A final 2 GPU hours were used for further hyperparameter tuning and model testing.

REFERENCES

[1] Fernández, Santiago, Alex Graves, and Jürgen Schmidhuber. ”Phoneme recognition in TIMIT with BLSTM-CTC.” *arXiv preprint arXiv:0804.3269* (2008). Table 2.