INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Text Augmentation for Language Models in High Error Recognition Scenario

Karel Beneš, Lukáš Burget

Brno University of Technology, Czechia
{ibenes,burget}@fit.vutbr.cz

Abstract

In this paper, we explore several data augmentation strategies for training of language models for speech recognition. We compare augmentation based on global error statistics with one based on unigram statistics of ASR errors and with label-smoothing and its sampled variant. Additionally, we investigate the stability and the predictive power of perplexity estimated on augmented data. Despite being trivial, augmentation driven by global substitution, deletion and insertion rates achieves the best rescoring results. On the other hand, even though the associated perplexity measure is stable, it gives no better prediction of the final error rate than the vanilla one. Our best augmentation scheme increases the WER improvement from second-pass rescoring from 1.1 % to 1.9 % absolute on the CHiMe-6 challenge.

Index Terms: data augmentation, error simulation, language modeling, automatic speech recognition

1. Introduction

The traditional reason language models (LMs) appear in ASR systems is that they directly represent the prior term $P(S)$ in the Bayes factorization of the posterior probability $P(S|A)$ of a sentence S given the audio A. However in practice, LMs trained on excessive amounts of data are combined with hybrid and end-to-end systems alike [1, 2, 3] at authors' liberty. Overall, LMs can be seen as a refinement tool to apply on a preliminary result of recognition.

To this end, language models can be trained to cope with errors introduced during the first phase of ASR. This is especially well pronounced in discriminative LMs [4, 5, 6], which focus on obtaining the final hypothesis from a pool of first-pass hypotheses, oftentimes explicitly taking acoustic clues into account. Errors are also freely introduced into LMs purposed to serve as a base for NLP systems [7]. On the other hand, we have recently achieved interesting improvements in ASR by simply augmenting the training data for a conventional generative LM [8].

In general, the idea of working with data similar to ASR output is not new: Literature typically focuses on errors introduced in the form of substitutions, driven by a custom *confusability* measure [9]. Traditionally, this confusability is based on phonemic confusions [10], Recently, sequence-to-sequence models have been proposed to introduce context-dependent errors [11].

In this work, we elaborate on the idea that an LM should be capable of good predictions of the next word even when it is exposed to some mistakes in the history: We extend the idea of augmenting data from substitutions only to deletions as well as insertions. We also explore how the augmentation effect changes when we remove the error statistics and do the augmentation in an uninformed manner. Then, we investigate the source of improvement by comparing this well-motivated input augmentation to target augmentation. Finally, we examine the impact of the data augmentation on the perplexity as a measure of LM quality.

2. Simulating the Errors

In the traditional setting, language models are trained to maximize the probability of the word w_t at any position t in the text, as conditioned on the history h_t comprising all previous words $w_1 \dots w_{t-1}$:

$$-\log \mathrm{PPL} = \frac{1}{T}\sum_{t=1}^{T} \log p(w_t|h_t) \tag{1}$$

In this study, we expose the LM to erroneous h_t, similar to what it experiences when processing output from an ASR system, giving rise to *simulated PPL*:

$$-\log \mathrm{sPPL} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{h}_t \sim p_{\mathrm{ASR}}(h_t)} \Big[\log p(w_t|\hat{h}_t) \Big] \tag{2}$$

We discuss the approximations of $p_{\mathrm{ASR}}(h_t)$ in Section 2.1. In general, we obtain $\hat{h}_t$ from the input history h_t by processing it token by token and introducing individual edits, as illustrated in Fig. 1. We take care not to remove, replace or introduce sentence boundaries[1].

We also introduce target augmentation to check that the LM benefits from modeling of ASR errors rather than simply from regularization by adding noise to the data. Target augmentation differs in that when introducing substitutions, we keep the input token and replace the target one. Contrasting this to the simulated perplexity (omitting deletions and insertions), we arrive at:

$$-\log \mathrm{tPPL} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{w}} \Big[\log p(\hat{w}_t|h_t) \Big] \tag{3}$$

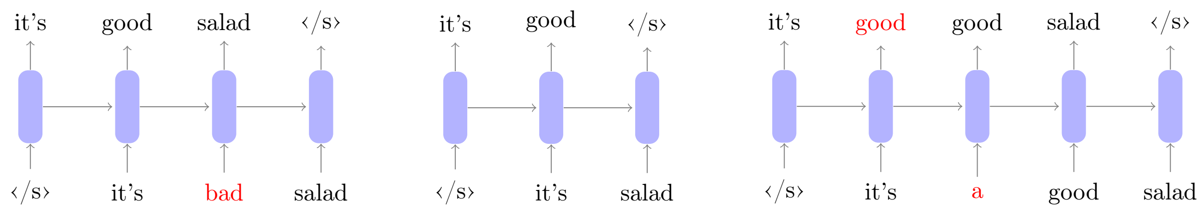
2.1. Error simulating distribution

As a baseline for error simulation, we do the sampling in a truly flat manner: We simply roll an unfair 4-sided dice to determine which of the four actions to take. In case substitution or insertion is selected, we take a sample from a uniform distribution over the vocabulary to obtain the new input token. We call this the *0-gram* error model. By adjusting the initial categorical distribution (the dice), we have a fine control over the strength of the data augmentation.

In order to better match the actual errors made by the first-pass recognizer, we propose a stronger, *1-gram*, model. We prepare it as follows: For a given set of utterances, we get the 100-best hypotheses from the ASR system. Secondly, we align these

[1] Due to a bug in the implementation, we provided no special care to sentence breaks initially. Despite being theoretically unsound, it did not have any observable impact.

  http://dx.doi.org/10.21437/Interspeech.2021-627



(a) *Substitution* (b) *Deletion* (c) *Insertion*

Figure 1: *How errors are introduced for NN LMs. The original sentence is* "it's good salad"*. With substitution (a), only the input token is replaced. To simulate deletions (b), both the input and the target at the given position are removed. Finally, inserted words (c) get into the input while the original target token gets duplicated.*

hypotheses to the actual transcriptions of this data. Then for each reference word w_{act}, we summarize its alignments over the whole data and normalize the counts of hypothesised words to get the distribution $p_{sub}(w|w_{act})$. This categorical distribution is then used to decide the action to apply on every training token. Note that by treating the empty symbol ε as a regular word, we naturally model insertions and deletions this way. In case of 1-gram error models, the overall rate of substitutions, deletions and insertions is given by the statistics themselves.

3. Experiments

We evaluate the proposed techniques on Track 1 of the CHiMe-6 challenge [12]. The size of training and development data, including sentence boundaries, is 522k and 136k tokens respectively. For ease of implementation, we stay with the official large vocabulary of 127k words, letting the output softmax layer to learn that many of those words do not occur in the training data.

As the ASR system to provide inputs for our experiments, we used a single Kaldi system based on a mix of CNN and TDNN-f layers. The first-pass decoding network is based on a KN-smoothed 3-gram LM. For further details on the design of the ASR system, refer to the system description [8].

This system achieves 48.39 % WER on the development data. The error is composed respectively of 5 %, 17 % and 26 % of insertions, deletions and substitutions.

For any LM evaluated, we extract 3000-best hypotheses to rescore and use the development set to tune the linear interpolation coefficient for mixing the LSTM LM with the first-pass 3-gram LM. When rescoring, we carry the hidden states over, except for session breaks. This way, we effectively model the language across segments [13].

3.1. Language Model Training

In all our experiments, we a use two layer LSTM [14] with 650 units per layer and the dimensionality of input word embeddings reduced to 100. We train our language models using BrnoLM[2].

We train the LMs with plain SGD, in two stages: At first, we train the LM from scratch with shuffled lines[3]. In this stage, we always employ the data augmentation technique under test. Once the perplexity stops improving[4], we take the trained LM and finetune it on the sentences in their original order. We ran this stage twice, with the augmentation turned either on or off. It was always slightly better to do this finetuning with clean data, thus we only report these results.

We begin the first phase training with learning rate 2.0 and start the finetuning with 0.2, in both stages halving the learning rate when development perplexity does not improve. With target augmentation, we observed the training to be more noisy, therefore we only halved the learning rate when no improvement was observed for 3 consecutive epochs.

3.2. Tested Augmentation Schemes

In total, we test LMs trained with seven augmentation schemes:

1. The *baseline*, which is only trained on the actual training transcripts.
2. The 0-gram model (i0), which is trained with uniformly sampled errors. With this model, we optimize for the best rate of substitutions, deletions and insertions.
3. The 1-gram model (i1), where we collect the statistics from the training data.
4. The oracle 1-gram model (i1o), where the statistics are collected from the development data.
5. A 0-gram target augmentation (t0S), where we only introduce substitutions, at the rate optimal for input augmenting systems.
6. A 0-gram target augmentation (t0SDI), where we introduce deletions and insertion in addition to the target substitution.
7. Target label smoothing (t0LS) [15]. Note that unlike the other techniques, label smoothing requires a principally different change of the training procedure.

For all augmentation schemes, we sweep across the rate of dropout in range [0.0, 0.7] to find the optimal level of total regularization. For the baseline, the best result comes from setting it to 0.7[5], those trained with data augmentation were fairly robust to the dropout rate and achieved their best performance in range of 0.3 – 0.6.

3.3. CHiMe-6 Rescoring Results

We first assess the performance on the development data, as captured in Figure 2. Overall, we see that the behavior of all LSTM LMs is smooth with respect to the interpolation coefficient, but the final WERs do differ significantly.

The best performance is achieved by the input 0-gram augmentation. For this augmentation, we have found values of

[2] https://github.com/BUTSpeechFIT/BrnoLM

[3] This effectively breaks across-segment dependencies.

[4] The models converged after around 40 epochs.

[5] In this case, we tried higher values to check it is the optimum.

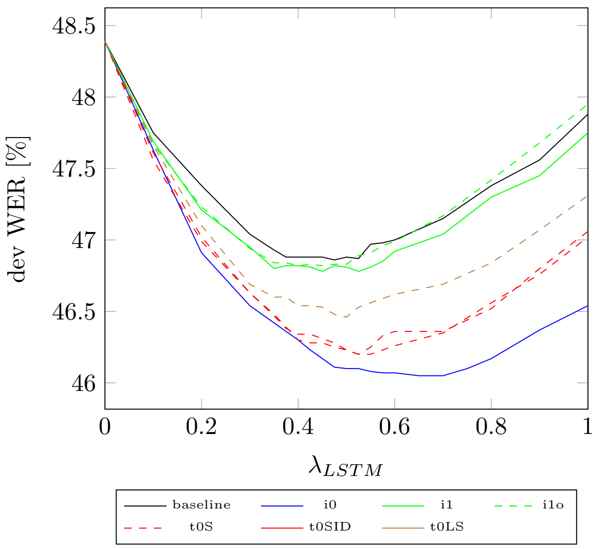


Figure 2: *Development WER as a function of LSTM LM weight. For each augmentation scheme, the best dropout rate is selected. The left edge represents retaining only the LM score from the original 3-gram LM. Note how – with the minor exception of oracle 1-gram input augmentation (i1o) – the performance of the LSTM LM itself (right edge) corresponds to the performance of its optimal interpolation.*

around 0.23, 0.15 to work the best as the substitution and deletion rate respectively. Insertions did not provide any measurable improvements up till 0.1; higher values caused degradation. Note that the optimal augmentation rates correspond rather closely to the actual errors rates of the ASR system (see Sec. 3)

On the other hand, the input 1-gram augmentation was not significantly better than the baseline. We ruled out several possible causes: This is not originating from the decreased *amount* of augmentation, as an i0 LM trained with the same amount of augmentation achieves 46.30 % WER. Also, we ruled out impact of outlier errors by limiting p_{ASR} so that it would only introduce errors that were observed at least five times in the training data. This brought no improvement. Neither is it because of a poor fit of the error statistics, as the 1-gram augmentation performs very similar even when based on oracle statistics. Therefore, it must be the nature of the augmentation that prevents gains. We assume that by being actually predictable, it produces little extra stimulus for learning.

The target augmenting LMs do achieve improvement over the baseline, albeit smaller than the 0-gram input augmentation. Introduction of deletions and insertions brought no improvement for the target 0-gram augmentation, however we have observed these LMs to be less sensitive to the dropout rate.

The two best performing schemes — input 0-gram and target 0-gram — are technically orthogonal to each other. Thus, we extended our experiments to an interpolation of the two: We kept the total substitution rate at 24 %[6] and varied the amount of input and target substitution. Unfortunately, no synergy was achieved and the development WER increased monotonically as more of the substitutions happened in the targets.

Finally, in Table 1, we capture the performance of the models on the unseen evaluation data. The input 0-gram augmentation brings clearly the largest gain and the overall behaviour of

[6] deletions at 15 % and insertions at 4 %

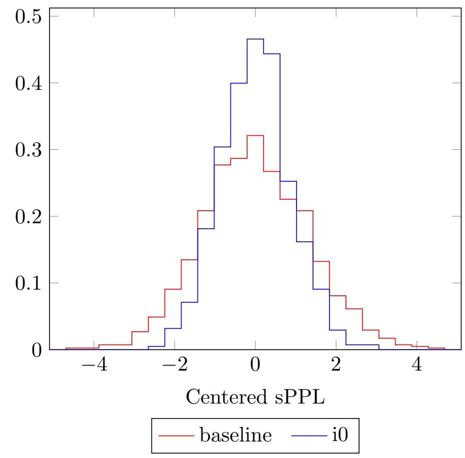


Figure 3: *Normalized histogram of sPPL estimates with substitution and deletion rate respectively 0.23 and 0.15. Estimated from 1000 runs across the development data. Note that values have been centered, baseline LM has mean-sPPL 226, the i0 one trained with matching data augmentation has 178.*

the other LMs is consistent with the development data. Only notable exception is the LM trained with label smoothing, which suddenly provides no benefit. We checked that this is not a result of mis-calibrated LM weight by tuning it to the optimal value w.r.t. evaluation data, but we did not find the cause of the difference.

Table 1: *Results of rescoring ASR outputs with LMs trained with different data augmentation schemes. We report the result of the optimal setting of dropout and LSTM-LM weight as per development results.*

	development	evaluation
3-gram only	48.39	48.82
baseline	46.86	47.69
input 0-gram	**46.05**	**46.92**
input 1-gram	46.85	47.70
input 1-gram oracle	46.82	47.97
target 0-gram S	46.20	47.41
target 0-gram SID	46.20	47.23
target label smoothing	46.43	47.70

3.4. Experiments on Firefighter Speech Recognition

To explore the efficiency of the proposed augmentation in a different scenario, we have applied the 0-gram input augmentation to speech recognition in the OpenSAT challenge [16]. This task is considerable easier, with WER at around 10 %. By adding the augmentation, we have achieved a marginal improvement of about 0.1 % absolute, confirming our expectation that this method brings benefit mainly when employed in high error scenarios.

3.5. Behavior of the Simulated Perplexity

Since the best performing LMs are trained with the sPPL objective, we investigate two of its properties. Firstly, we observe its

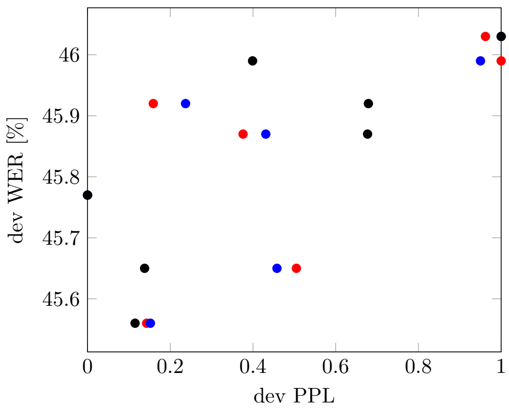


Figure 4: *Correlation of different PPL estimates with development WER. Individual points represent LMs trained with different rate of i0 augmentation. Red, blue and black denote respectively sPPL following development error rates, sPPL following training error rates and the vanilla PPL. For each of the PPL variants, the measured PPL values have been min-max normalized.*

stability as we only estimate the expectation in (2) from a single realization of noise. In doing so, we compare behavior of two different LMs, where one was trained to optimize sPPL and the other was optimized with regular PPL. Then, we examine the predictive power of sPPL towards the final WER of the system.

The stability is captured in Figure 3. The sPPLs are approximately normally distributed, with relative standard deviation of around 0.5 %. Comparing the baseline LM to the one trained on the augmented data, we see that the baseline has significantly higher average sPPL (226.5 vs. 178.7) and a slightly higher standard deviation.

To assess the predictive power of sPPL, we plotted a couple of i0 LMs as described by their development (s)PPL and WER in Figure 4. We did not find any conclusive evidence that sPPL would serve as a better predictor than PPL.

4. Conclusions

We have examined several simple text data augmentations for language model training. Evaluating them by rescoring ASR outputs on the CHiMe-6 challenge, we have achieved the best result when simply introducing uninformed edits into the stream of input tokens. This improved the WER by 0.8 % absolute over rescoring with the baseline LM, which had the same neural architecture, but was trained on clean text only. No improvements were achieved with augmentation based on the actual word level confusions produced by the ASR system. Finally, a control experiment with target augmentations reached approx. 0.5 % abs. improvement. From these experimental results, we conclude that while the LMs do benefit from being exposed to ASR-like errors, most of the improvement is coming from training on noised data per se.

We also investigated into the properties of simulated perplexity (sPPL) estimated on the augmented data. While being rather stable w.r.t. the sampling of the word-level noise, we did not see sPPL predict the WER any better than PPL. Furthermore, we always obtained better results when finetuning the LMs to clean data, solidifying our belief that the proposed augmentation scheme should be viewed as a successful regularization technique rather than as an adaptation to a given ASR system.

In future, we will focus on explaining the failure of the augmentation based on actual error statistics and we will seek a remedy to it.

5. Acknowledgements

The work was supported by Czech National Science Foundation (GACR) project "NEUREM3" No. 19-26934X. Part of high-performance computation run on IT4I supercomputer and was supported by the Ministry of Education, Youth and Sports of the Czech Republic through e-INFRA CZ (ID:90140). We thank Martin Kocour for providing the CHiMe-6 ASR system outputs.

6. References

[1] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Language Modeling with Deep Transformers," in *Proc. Interspeech 2019*, 2019, pp. 3905–3909. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2225

[2] S. Toshniwal, A. Kannan, C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2018, pp. 369–375.

[3] J. Cho, S. Watanabe, T. Hori, M. K. Baskar, H. Inaguma, J. Villalba, and N. Dehak, "Language model integration based on memory control for sequence to sequence speech recognition," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 6191–6195.

[4] B. Roark, M. Saraclar, and M. Collins, "Discriminative n-gram language modeling," *Computer Speech & Language*, vol. 21, pp. 373–392, 04 2007.

[5] Z. Zhou, J. Gao, F. Soong, and H. Meng, "A comparative study of discriminative methods for reranking LVCSR n-best hypotheses in domain adaptation and generalization," vol. 1, 06 2006, pp. I – I.

[6] T. Oba, T. Hori, A. Nakamura, and A. Ito, "Round-robin duel discriminative language models," *IEEE Transactions on Audio, Speech & Language Processing - TASLP*, vol. 20, pp. 1244–1255, 05 2012.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[8] K. Žmolíková, M. Kocour, N. F. Landini, K. Beneš, M. Karafiát, K. H. Vydana, A. D. Lozano, O. Plchot, K. M. Baskar, J. Švec, L. Mošner, V. Malenovský, L. Burget, B. Yusuf, O. Novotný, F. Grézl, I. Szőke, and J. Černocký, "BUT system for CHiME-6 challenge," in *Proceedings of CHiME 2020 Virtual Workshop*. University of Sheffield, 2020, pp. 1–3. [Online]. Available: https://www.fit.vut.cz/research/publication/12283

[9] E. Simonnet, S. Ghannay, N. Camelin, and Y. Estève, "Simulating ASR errors for training SLU systems," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. [Online]. Available: https://www.aclweb.org/anthology/L18-1499

[10] P. Jyothi and E. Fosler-Lussier, "Discriminative language modeling using simulated asr errors," 01 2010, pp. 1049–1052.

[11] P. Serai, P. Wang, and E. Fosler-Lussier, "Improving speech recognition error prediction for modern and off-the-shelf speech

recognizers," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7255–7259.

[12] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj, D. Snyder, A. S. Subramanian, J. Trmal, B. B. Yair, C. Boeddeker, Z. Ni, Y. Fujita, S. Horiguchi, N. Kanda, and T. Yoshioka, "CHiME-6 Challenge: Tackling multispeaker speech recognition for unsegmented recordings," in *6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, 2020.

[13] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Training language models for long-span cross-sentence evaluation," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 419–426.

[14] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *INTERSPEECH*, 2012.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[16] M. Karafiát, M. K. Baskar, I. Szöke, H. K. Vydana, K. Veselý, and J. H. Černocký, "BUT Opensat 2019 speech recognition system," 2020.