WARPED LANGUAGE MODELS FOR NOISE ROBUST LANGUAGE UNDERSTANDING

Mahdi Namazifar, Gokhan Tur, Dilek Hakkani-Tür

Amazon Alexa AI

ABSTRACT

Masked Language Models (MLM) are self-supervised neural networks trained to fill in the blanks in a given sentence with masked tokens. Despite the tremendous success of MLMs for various text based tasks, they are not robust for spoken language understanding, especially for spontaneous conversational speech recognition noise. In this work we introduce Warped Language Models (WLM) in which input sentences at training time go through the same modifications as in MLM, plus two additional modifications, namely inserting and dropping random tokens. These two modifications extend and contract the sentence in addition to the modifications in MLMs, hence the word "warped" in the name. The insertion and drop modification of the input text during training of WLM resemble the types of noise due to Automatic Speech Recognition (ASR) errors, and as a result WLMs are likely to be more robust to ASR noise. Through computational results we show that natural language understanding systems built on top of WLMs perform better compared to those built based on MLMs, especially in the presence of ASR errors.

1. INTRODUCTION

Masked Language Models (MLM) and specifically BERT which were introduced in [1] are trained by masking some tokens of the input and predicting what the original masked tokens are. The masking of tokens in MLMs is inspired by the Cloze task [2] in which participants are asked to guess the missing/masked words in sentences. Even though predicting masked tokens, which is referred to as MASK, is the main task in BERT, there are two additional tasks that BERT is trained on. These two tasks are predicting that the token in a given position is kept intact (referred to as KEEP), as well as predicting the original token at a specific position in the input sentence that is substituted by another random token (referred to as RAND). Therefore one could think of BERT as a multi-task model. A natural consequence of thinking of BERT as a multi-task model is the idea of adding even more tasks to the list of tasks that BERT is trained to solve in order to achieve a richer language model with a more diverse capabilities, or even making the process of training these language models more efficient. In this work we introduce two additional tasks to the set of tasks that BERT is trained on, namely we

- insert a random token in a random position in the input sentence and train the model to predict a special token for those positions, which we refer to as INSERT.

- delete the token at a random position in the input sentence and train the model to predict what the deleted token is, which we refer to as DROP.

We refer to these new language models as Warped Language Models (WLM) since in the training process the input text is first "warped"
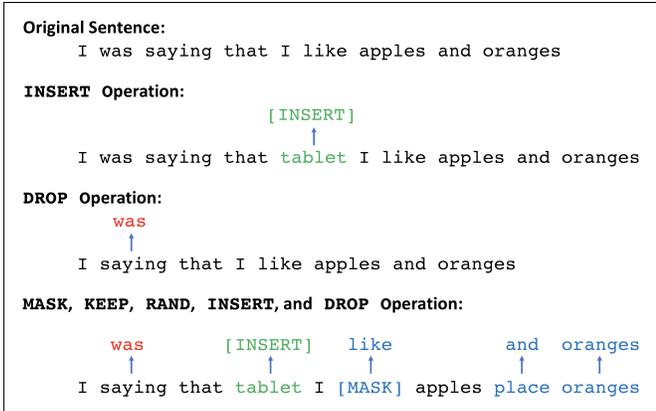
```
Original Sentence:
    I was saying that I like apples and oranges

INSERT Operation:
                        [INSERT]
                           ↑
    I was saying that tablet I like apples and oranges

DROP Operation:
        was
         ↑
    I saying that I like apples and oranges

MASK, KEEP, RAND, INSERT, and DROP Operation:

        was        [INSERT]    like            and   oranges
         ↑             ↑          ↑              ↑       ↑
    I saying that tablet I [MASK] apples place oranges
```

Fig. 1: Example of warping in WLMs. The inserted tokens get the special insert token as their label (shown in green). The tokens following a deleted token get the deleted token as their label (shown in red). The `MASK`, `KEEP`, and `RAND` operations from original BERT are shown in blue.

(masked, extended, contracted, etc.) through the operations mentioned above, namely some tokens are masked, some tokens are substituted by other random tokens, some tokens are deleted, and some random tokens are inserted in random positions.

The noise that is introduced by these warping operations, among other things, forces the language model to be more robust to input noise. As BERT and WLM are based on transformers [3] which themselves heavily rely on self-attention, they pay attention to the entire input even in the presence of noise and errors. Such input noise is specifically inherent in Spoken Language Understanding (SLU) tasks that as input use the output of an Automatic Speech Recognition (ASR) system. ASR noise is quite similar to the noise that is introduced by the warping operations `RAND`, `INSERT`, and `DROP` in nature. More specifically, the ASR output might have some words of the spoken sentences substituted by other words, some words that are inserted into the spoken sentence, or some words of the spoken sentence that are deleted. In this work we focus on the impact of the additional two tasks that we introduce (`INSERT` and `DROP`) on SLU systems built on top of WLMs.

In our experiments through evaluation of Spoken Language Understanding (SLU) systems

trained on the Airline Travel Information System (ATIS) data set [4, 5, 6] we show that SLU systems built on top of WLMs are more robust to ASR noise compared to those built on top of MLMs, and specifically BERT.

2. RELATED WORK

Following the success of BERT, GPT-2 [7], and ELMo [8], that used masking or next token prediction as training tasks, a number of other works have looked into different tasks as basis for training language models. For instance instead of masking the inputs ELECTRA [9] substitutes some of the tokens with tokens sampled from a generator network and a discriminator detects what token was substituted. In BART [10] the authors train a denoising auto-encoder where the noising function corrupts the input through token masking, token deletion, sentence span masking, sentence permutation and document rotation. Token deletion in our WLM is different from token deletion in BART since in BART the model decides which positions (i.e., word boundaries) are missing inputs whereas in WLM the model predicts the missing token for the token that comes after it. In another recent work [11] the task based on which the language model is trained is a multi-choice Cloze task with "reject" as one of the choices. In this work a separate network provides training input and choices.

Regarding the applications of WLMs that are of interest in this work, namely SLU from ASR output, several recent works have looked into creating models that are robust to ASR noise. In [12] the authors use an ASR simulator to inject noise in the otherwise error-free training data for SLU tasks. In a different yet related approach in [13] a "confusion" loss is added to the language model's training loss to force it to produce similar embeddings for acoustically confusable words. In [14] the authors propose a joint model to both correct ASR output and language understanding using the ASR n-best hypotheses. There are also several works that use transformer

[3] based language models to improve ASR output quality. [15] trains a transformer based encoder-decoder architecture to translate ASR output into semantically and grammatically correct text. In [16] the authors use transformer-based language models for lattice rescoring [17].

3. WARPED LANGUAGE MODELS

As we described in the Introduction section, we define WLMs to train on two additional task on top of the `MASK`, `KEEP`, and `RAND` tasks that are used in BERT. These two additional tasks are `INSERT` and `DROP`. In the next two subsections we describe the details of these two tasks and Figure 1 depicts examples of each task.

3.1. `INSERT`

In this task random tokens are inserted at random positions in the input text. For these positions the language model is trained to predict a special token that is only used for indicating an inserted token. In other words during training the inserted tokens get the `INSERT` special token as label. The embedding for this special token is randomly initialized at the beginning of the training process, and since it does not appear in the training data, there is no back-propagation into its embeddings and as a result, this embedding remains intact throughout the training process, while all other token embeddings continue to change by the back-propagation process.

3.2. `DROP`

In the `DROP` task, during the training of the WLM, some tokens are randomly deleted from the input sentences. For each deleted token the model is forced to predict the deleted token for its subsequent token. In other words, during training of the WLM, the token following a deleted token gets the deleted token as its label.

3.3. Masking vs Warping

The operations that are applied on the input task during training of MLMs, namely `MASK`, `KEEP`, and `RAND` do not change the length or the structure of the input sentences. These operations could simply be implemented using element-wise operations of an operating vector (a vector that indicates where to `MASK`, `KEEP`, or `RAND`). One could think of this operating vector applied element-wise on the input tokens vector as performing "masking" on the input, as it is a layer that sits on top of the input vector. On the other hand, the two additional operations that we introduce in WLMs, namely `INSERT` and `DROP` change the length and structure of the input sentence. These two operations cannot be implemented by simply applying an operating vector to the vector of the input tokens, as is the case for the operations in MLMs.

The two new operations `INSERT` and `DROP` are fundamentally different from the original operations of MLMs (`MASK`, `KEEP`, `RAND`). The operation `INSERT` forces the model to return a fixed token every time it determines that a token was inserted into the input sentence, given the context. The operation `DROP` is the more bizarre of the two since it anchors the sentence at the positions followed by a deleted token to the previous position by labeling the token after a deleted token with the deleted token. In all of the other operations in WLMs, the label at a position is directly indicating what the token in that position should be (in the case of `INSERT` the token should be none), but in the `DROP` operation the label is indicating what the previous token should be. Intuitively this operation might seem destructive to the training process, but as we show in the experimental results, the combination of the two new operations not only does not impair the convergence of the language model, it in fact improves the performance of the language model on downstream SLU tasks involving ASR noise.

Finally it should also be noted that WLMs are generalization of MLMs where the `INSERT` and

DROP are introduced in addition to MASK, KEEP, and RANDoperations at training time. Note that the difference between WLMs and MLMs are only in how they are trained and beyond that, the architecture, input and output formats, etc. are identical.

3.4. Illegal Operations

Another way that the two operations that we introduce in this paper (INSERT, DROP) are different from MLM's original operations (MASK, KEEP, RAND) is that the these new operations could create warping of sentences that should not be permitted. The first case that creates an illegal warping is where a MASK, KEEP, or RAND operation appears immediately after a DROP. This is an illegal case because if it happens the token after the deleted token would get two different labels, one of which would be the deleted token for the DROP operation and the other one would be the label associated with the MASK, KEEP, or RAND operation. The second case that creates an illegal warping is when more than one consecutive DROP operations happen. In this case there would be no way to indicate the first deleted token as a label. In the implementation of WLMs we make sure that these two illegal cases do not occur during the training process.

4. EXPERIMENTS

We implemented WLM by generalizing the BERT implementation within the XLM framework [18] to cover INSERT and DROP operations. We then train a BERT model as well as a WLM model for English using the entire Wikipedia data from scratch, to use as a baseline. For these models we used 512 for the embedding dimensions and a 12 layer transformer encoder module with 16 attention heads. In this setup the model has about 56 million parameters. We trained these models (WLM and BERT) using 8 GPUs on an Amazon AWS p3.16xlarge instance for 100 epochs. The training process takes over 4 days. In the original BERT paper the authors

propose to first randomly pick 15% of tokens, and then among those tokens MASK 80%, KEEP 10%, and RAND 10% of them. We follow this exact split for training BERT. However for training WLM, we first randomly pick 15% of tokens (similar to BERT), but then among those tokens we MASK 60%, KEEP 10%, RAND 10%, INSERT 10%, and DROP 10%.

Table 1 shows perplexity and accuracy of WLM and BERT after 100 epochs of training measured on the same validation and test sets. From these numbers we can see that the final BERT model (after 10 epochs of training) has a slightly better perplexity and accuracy on the test sets. Note the the architecture and the number of parameters of both models are identical.

	Validation		Test	
	Perp.	Acc.	Perp.	Acc.
WLM (56M)	7.68	59.61	7.80	59.01
BERT (56M)	7.34	60.22	7.43	59.72

Table 1: Perplexity and Accuracy of WLM and BERT after 100 epochs of training. Both models have approximately 56 million parameters. Accuracy numbers are computed identically for WLM and BERT

4.1. WLM for SLU with ASR Noise

Since WLM encounters INSERT and DROP operations during its training we expect that it would be more robust to these types of noise in the input text. These types of noise are frequently seen in ASR outputs. In this section through our experiments we show that SLU systems built for ASR outputs that use WLM perform better than the same models that use BERT instead of WLM.

4.1.1. Data: ATIS with ASR Noise

The ATIS dataset [4, 5, 6] is a widely used benchmark for natural language understanding and provides audio recordings as well as manual transcriptions of utterances inquiring a flight

information system. The ATIS dataset also provides manual labels for the 81 slots and 18 intents that appear in the data. In this paper, we use the ASR output for this dataset that was released by Huang and Chen [13]. We aligned these ASR outputs with the manual transcripts using SCTK, the NIST scoring toolkit [1]. We obtained the slot tags for the ASR output by transferring the labels from manual transcriptions to aligned words in the ASR output and updating the labels to preserve the IOB-style annotation [19].

The original ATIS dataset release does not contain a validation set, hence, we used the training, validation, and test set split from [20], which includes 4478, 500, and 893 utterances in the training, validation and test subsets, respectively. Table 2 shows the word error rate (WER) as well as insertion (INS), deletion (DEL), and substitution (SUBS) error rates in the training, validation, and test subsets.

Subset	WER	INS	DEL	SUBS
Train+Val	18.6%	3.3%	2.4%	12.9%
Test	15.9%	2.9%	1.5%	11.5%

Table 2: Word error rate (WER), insertion (INS), deletion (DEL), and substitution (SUBS) error rates in the training, validation, and test subsets of the ATIS ASR outputs released by [13].

4.1.2. SLU Model

We build our SLU models with WLM and BERT following the general architecture proposed in [21] to jointly predict intents and slots on top of WLM and BERT as shown in Figure 2. More specifically we add an intent prediction head and slot prediction heads to the output of WLM and BERT. We use the ATIS training set to train this model by back-propagating the loss through the entire model (intent classification head, slot tagging heads, and the language model).
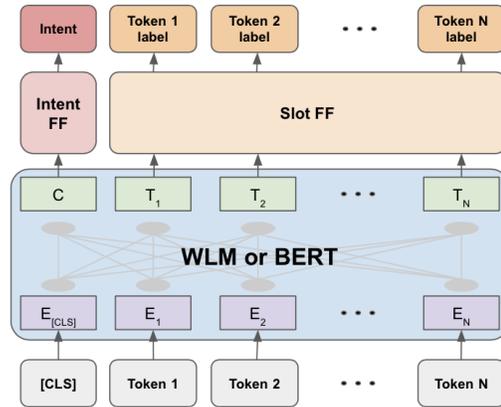
Fig. 2: SLU model following the architecture proposed in [21]. "FF" in the boxes above the BERT box stands for feed forward network.

4.1.3. Evaluation

For reporting performance, we use three measures: intent accuracy (the percentage of utterances with correct intents), slot F1 score (CoNLL Style, [22]), and sentence level joint accuracy (the percentage of utterances which have all the slots and intents accurately annotated).

4.1.4. Experimentation Results

In our experiments, as a result of our aligning transcribed and ASR data, for each of the training and test sets we have the intents and slots labels for both transcribed and ASR data. Therefore we use the following three settings for our experiment:

- Train on transcribed data and test on transcribed data (Transcribed–Transcribed)

- Train on transcribed data and test on ASR data (Transcribed–ASR)

- Train on ASR data and test on ASR data (ASR–ASR)

For each of these settings we run the training process 5 times and the numbers that we report are averaged over these 5 runs. Table 3 summarizes the results of these experiments. Numbers in bold

	Train on Transcribed Test on Transcribed			Train on Transcribed Test on ASR			Train on ASR Test on ASR		
	intent	slot	joint	intent	slot	joint	intent	slot	joint
WLM (56M)	97.91	93.65	84.23	93.93	**84.14**	**59.59**	95.51	**88.72**	**70.10**
BERT (56M)	97.77	93.41	83.18	93.69	83.08	58.25	95.28	87.39	67.89

Table 3: Performance of the SLU model built on top of WLM and BERT, where both WLM and BERT models have around 56 million parameters. Values for "intent" are accuracy of intent detection, values for "slot" are F1 measure of slot detection, and, finally, values for "joint" are joint accuracy which is the percentage of utterances which have all slots and intents correctly annotated. Bold values represent statistically significant difference at p-value of 0.05.

show the values that are higher in a statistically significant manner at p-value of 0.05. From the table we could see that the difference between intent accuracy for WLM and BERT in all of the three settings is negligible. This could be due to the fact that ASR noise does not interfere with intent detection significantly since, intuitively, even in presence of some noise, detecting the intent of an utterance is still possible. Also the difference between WLM and BERT for slot detection in the case where both training and testing is done on transcribed data is notably negligible. The differences between slot detection F1 scores and joint accuracy measures become more prominent when we consider the cases where both training and testing or only testing is done on the ASR data. In both these cases it is clear from the table that the models perform significantly worse overall due to the presence of ASR noise in either the test set only or in both training and test sets. This drop in performance is more pronounced for the case where training is done on transcribed data and testing is done on ASR data, which could be explained by distribution mismatch between training and test sets. In this case, we see that WLM improves the performance on slot detection and joint accuracy significantly. This improvement is also clear in the case where both training and testing is done on the ASR data, where we see that the joint accuracy of the BERT based model is 67.89, whereas for the WLM based model it is 70.10.

For the case where both training and testing is done on the transcribed data the slot F1 measure (93.65) is lower than the numbers reported for BERT-base in [21] (95.59). But note that BERT-base model has around 110 million parameters, but we trained our BERT and WLM with 56 million parameters due to the long time it takes to train these models (over 4 days on an 8 GPU machine).

5. CONCLUSIONS

In this work we introduce WLMs, that are generalization of MLMs, through introduction of two additional training tasks, namely `INSERT` and `DROP` where the model is trained to predict where a random token is inserted into or deleted from the input sequence during training. Due to the new noise that WLMs see at training time, they are more robust to noisy input. We demonstrate this by showing that SLU models built on WLMs perform better than the same models built on MLMs when ASR noise is present.

There is however much more to be studied regarding WLMs to better understand their strengths and weaknesses. Moreover, there are other behaviors of WLMs that are not understood to us. For instance the fact that in the `DROP` operation the label of tokens come from the previous position does not throw off the training process needs to be studied in more depth. Moreover, the fact that there is no distinguishing factor for the model between the label of a token following a `DROP`ed token and the label of a `RAND` token, and yet somewhat surprisingly the training process still converges, needs to be studied more.

6. REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina 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, June 2019, pp. 4171–4186, Association for Computational Linguistics.

[2] Wilson L. Taylor, ""cloze procedure": A new tool for measuring readability," *Journalism Quarterly*, vol. 30, no. 4, pp. 415–433, 1953.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., pp. 5998–6008. Curran Associates, Inc., 2017.

[4] Charles T. Hemphill, John J. Godfrey, and George R. Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*, 1990.

[5] Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg, "Expanding the scope of the ATIS task: The ATIS-3 corpus," in *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994.

[6] G. Tur, D. Hakkani-Tür, and L. Heck, "What is left to be understood in atis?," in *2010 IEEE Spoken Language Technology Workshop*, 2010, pp. 19–24.

[7] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, "Language models are unsupervised multitask learners," 2019.

[8] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, "Deep contextualized word representations," in *Proc. of NAACL*, 2018.

[9] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020.

[10] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer, "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 7871–7880, Association for Computational Linguistics.

[11] Zhenhui Xu, Linyuan Gong, Guolin Ke, Di He, Shuxin Zheng, Liwei Wang, Jiang Bian, and Tie-Yan Liu, "Mc-bert: Efficient language pre-training via a meta controller," 2020.

[12] Longshaokan Wang, Maryam Fazel-Zarandi, Aditya Tiwari, Spyros Matsoukas, and Lazaros Polymenakos, "Data augmentation for training dialog models robust to speech recognition errors," in *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, Online, July 2020, pp. 63–70, Association for Computational Linguistics.

[13] C. Huang and Y. Chen, "Learning asr-robust contextualized embeddings for spoken language understanding," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8009–8013.

[14] Y. Weng, S. S. Miryala, C. Khatri, R. Wang, H. Zheng, P. Molino, M. Namazifar, A. Papangelis, H. Williams, F. Bell, and G. Tur, "Joint contextual modeling for asr correction and language understanding," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6349–6353.

[15] O. Hrinchuk, M. Popova, and B. Ginsburg, "Correction of automatic speech recognition with transformer sequence-to-sequence model," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7074–7078.

[16] Kazuki Irie, Albert Zeyer, Ralf Schlüter, and Hermann Ney, "Language modeling with deep transformers," in *INTERSPEECH*, 2019.

[17] Martin Sundermeyer, Zoltán Tüske, Ralf Schlüter, and Hermann Ney, "Lattice decoding and rescoring with long-span neural network language models," in *INTERSPEECH*, 2014.

[18] Guillaume Lample and Alexis Conneau, "Cross-lingual language model pretraining," *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[19] Lance Ramshaw and Mitch Marcus, "Text chunking using transformation-based learning," in *Third Workshop on Very Large Corpora*, 1995.

[20] Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu, "Joint slot filling and intent detection via capsule neural networks," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[21] Qian Chen, Zhu Zhuo, and Wen Wang, "BERT for joint intent classification and slot filling," *CoRR*, vol. abs/1902.10909, 2019.

[22] Erik F. Tjong Kim Sang and Fien De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition," in *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, 2003, pp. 142–147.