# Comparison of Unsupervised Data Augmentation with BERT and XLNet

Thomas P. Goter

University of California, Berkeley
tomgoter@berkeley.edu

December 3, 2019

**Abstract**

*The Unsupervised Data Augmentation (UDA) methodology [1] is further extended for use with the XLNet transformer-based neural language model [2] for the purposes of comparing on a novel text classification dataset. The results discussed herein show that the benefits of UDA are reproducible and extensible to other modeling architectures; however, UDA is not a replacement for acquiring additional labeled data. Rather it is shown to be simply a performance-booster.*

## I. Introduction

The state-of-the-art in natural language processing and understanding (NLP/NLU) has been ever improving, especially in the last few years with the development of attention-based neural models [3] such as BERT [4] and XLNet [2]. While state-of-the-art models are continually being improved, there is still a need in many NLP/NLU tasks to obtain a large set of labeled training data for supervised finetuning. While this is an obvious observation, it is not necessarily so obvious what can be done when significant amounts of training data are available but much is unlabeled. This is why the recent work on UDA [1] is so exciting; it provides an opportunity to maximize the worth of unlabeled training data through a semi-supervised learning procedure.

In [1], UDA was found to be competetive with state-of-the-art, fully supervised neural models for binary text classification problems; however, for multi-class problems the same was not true. In other words, reducing training data, even when applying UDA, led to significant increases in error rate. Until this point the UDA methodology has only been coupled with the BERT architecture which was likely state-of-the-art during the development of UDA. Thus, the question of interest is whether UDA when applied to current state-of-the-art model architecture (i.e., XLNet) still provides reductions in error rate akin to what was shown in [1] when using BERT, and if so, are the error rate reductions significant enough to overcome reductions in labeled data. The ultimate goals of the work dicussed herein are to quantify, for a five-class text classification task, the following:

- Baseline classification performance on novel dataset and how performance varies with labeled data quantity
- Finetuned BERT and XLNet performance on the same dataset over the same range of dataset sizes
- Performance worth of UDA for BERT and XLNet based models

Here the UDA metric "performance worth" will be measured in two ways: 1) absolute error reduction rate and 2) absolute error reduction rate relative to that coming from increasing the labeled training set size.

$$PW = \frac{ER_{FT} - ER_{UDA}}{ER_{FT}} \tag{1}$$

The classification task chosen is to identify which novel from the *Song of Ice and Fire* series did a given sequence of text originate. The generation of this dataset and the specifics of this task are discussed in the following sections.

# II. Background

## i. Data

The dataset chosen for this multi-class text classification task is novel and was derived from the entire volume of the *Song of Ice and Fire* (a.k.a. *Game of Thrones*) novels by author, George R. R. Martin. This was chosen for two reasons 1) this text was expected to provide a challenging classification dataset due to the presence of many out-of-vocabulary words associated with the rich world created by Martin and 2) the author's personal interest and familiarity with this text.

The text of the novels was processed by the author and separated into *word* sequences[1] approaching lengths of 100. This length cap was chosen for three reasons 1) provide a significant number of training examples, 2) provide significant context in each training example from which to get value from the BERT and XLNet neural models, and 3) due to hardware limitations, specifically memory, it was expected that tokenized sequences would need to be capped at 128 based on the work discussed in [2] and [4][2]. Each sequence was labeled with which book (i.e., 1 through 5) in which is was found. A brief description of the dataset is provided as Table 1 which indicates there are a total of 19,438 examples in this new dataset.

Data was randomly shuffled and separated into training, development and test sets as shown in Table 2. Training data was further subdivided into smaller balanced, labeled subsets of size 20, 200, 2000, 5000, and 12000 for use in various studies discussed in Sections III and IV. Further as discussed in Section ii augmented datasets using all available training data (i.e., 15001 examples) were also generated for the UDA studies.

[1]Tokenization was done after generating training, development and test sets. Thus, the 100 word sequence limit left some room for expected increases in sequence length after tokenization and minimizing data truncation.

[2]Guidance on sequence length versus GPU/TPU memory availability is provided on the GitHub repositories associated with each of these articles.

**Table 1:** *Descriptive Statistics of Song of Ice and Fire Dataset*

| Title | Seqs | Median Length | Length Range |
|---|---|---|---|
| *Game of Thrones* | 3313 | 86 | [80-146] |
| *Clash of Kings* | 3608 | 86 | [54-157] |
| *Storm of Swords* | 4575 | 86 | [77-157] |
| *Feast for Crows* | 3307 | 86 | [74-209] |
| *Dance with Dragons* | 4635 | 86 | [80-144] |
| Totals | 19438 | 86 | [54-209] |

**Table 2:** *Division of Dataset*

| Dataset | Training | Development | Test |
|---|---|---|---|
| Examples | 15001 | 2501 | 1938 |

## ii. Methods

In this section, the methods for the baseline model, finetuned, fully supervised and UDA models (including data augmentation methodology) are briefly reviewed.

### ii.1 Baseline Model

The baseline model chosen for this multi-class text classification task was a simple Naive-Bayes model based. Unigram and bigram models were evaluated with simple count and tf-idf vectorizers. This simple model was chosen as the baseline for two reasons.

- Many of the sequences were assumed to be readily identifiable based on which characters and places appeared in a given sequence. Thus, a simple Naive-Bayes model would likely be effective for these passages.
- Show that the classification task is non-trivial. The baseline was simple enough such that if it achieved high accuracy independent of training set size, the task would have been identified as trivial and a different classification task would have been pursued. As will be shown in Section III, this was not the case, and this classification problem is indeed non-trivial.

**Table 3:** *Tokenization of Examples*

| Model | Example[3] |
|---|---|
| **Raw** | "Is it his fault the old man died? Stannis glanced into the fire. I never wanted Cressen at that feast. He'd angered me, yes, he'd given me bad counsel, but I did not want him dead. I'd hoped he might be granted a few years of ease and comfort. He had earned that much, at least, but—he ground his teeth together—but he died. And Pylos serves me ably. Pylos is the least of it. The letter . . . What did your lords make of it, I wonder?" |
| **BERT Tokenization** | [CLS] is it his fault the old man died ? stan ##nis glanced into the fire . i never wanted cr ##ess ##en at that feast . he ' d angered me , yes , he ' d given me bad counsel , but i did not want him dead . i ' d hoped he might be granted a few years of ease and comfort . he had earned that much , at least , but — he ground his teeth together — but he died . and p ##yl ##os serves me ab ##ly . p ##yl ##os is the least of it . the letter . . . what did your lords make of it , i wonder ? [SEP] [PAD] [PAD] [PAD] [PAD] [PAD] |
| **XLNet Tokenization** | "<unk> <unk> _I s _it _his _fault _the _old _man _died ? _Stan nis _glanced _into _the _fire . _I _never _wanted _Cre s sen _at _that _feast . _He ' d _angered _me , _yes , _he ' d _given _me _bad _counsel , _but _I _did _not _want _him _dead . _I ' d _hoped _he _might _be _granted _a _few _years _of _ease _and _comfort . _He _had _earned _that _much , _at _least , _but — he _ground _his _teeth _together — but _he _died . _And _Py los _serves _me _ ably . _Py los _is _the _least _of _it . _The _letter _ . _ . _ . _What _did _your _lord s _make _of _it , _I _wonder ? <sep> <cls>" |

### ii.2 Finetuned BERT Model

The finetuning of a BERT model for text classification is a relatively straightforward task in which existing model parameters are the learned embedding for the special [CLS] token is fed to an output layer softmax. The finetuning begins with using learned model parameters from an existing model [4], and weights from all layers are adjusted during the finetuning process. For the sake of this task, the BERT-base, uncased model (12-layer, 768 hidden layer dimension, 12 attention heads and 110M model parameters) has been used as the starting point. The learning rate should be set to avoid "catastrophic forgetting" of the pretrained weights by keeping the learning rate relatively small, but it is treated as a hyperparameter (along with attention and hidden layer dropout rates) during the finetuning process.

The BERT model uses a special tokenization that includes WordPiece tokenization [5] which breaks words down into smaller pieces and uses special # tokens to indicate when this word trimming occurs. This helps both with identifying similar root words, but also helps to accomodate out-of-vocabulary words. Table 3 compares an example sequence along with its BERT and XLNet tokenization. Code for the BERT model was leveraged from that used for the original UDA paper [1]; however, it was updated to be compatible with the Google Colaboratory environment (notebook-based), Tensorflow 1.15.0 and Python 3.

### ii.3 Finetuned XLNet Model

Using XLNet for text classification is quite similar to that for BERT. At the time of this analysis, only the cased XLNet models are available for download. This model has 117M parameters and is built on the TransformerXL model architecture [6]. As discussed in [2], the XLNet model uses an autoregressive method estimate a probability distribution of a word given not just its forward or backward context, but with respect to all possible permutations of context for a given sequence. This approach

was shown to outperform BERT specifically for text classification making it a naturally desirable model to apply to the classification task discussed herein. As with BERT, finetuning is performed by making small adjustments to the pretrained weights for all layers.

The XLNET model is distributed with an existing SentencePiece model (as BERT is distributed with a vocabulary) which is used for tokenization of sequences. Resultant tokenization differences are shown in Table 3. These differences are the explict (XLNet) versus implict (BERT) treatment of whitespace, the casing, how out-of-vocabulary words get broken down into wordpieces (e.g., p ##yl ##os and _Py los for the name "Pylos"), and ordering of special tokens in the sequence. Finetuning of model weights and hyperparameter tuning (i.e., learning rates, dropout, steps, et cetera) is all achieved with evaluation on the development set.

### ii.4 Unsupervised Data Augmentation

The goal of the work discussed herein was to compare the UDA methodology from [1] when coupled with BERT and XLNet models for this *Game of Thrones* text classification problem. As such the UDA methodology was kept consistent with that discussed in detail in [1]. This is a semi-supervised, consistency training method in which both labeled and unlabeled data is leveraged. In this process a training batch consists of three parts 1) labeled non-augmented examples $E_L$, 2) unlabeled, non-augmented examples $E_U$, and 3) matching unlabeled, augmented examples $E'_U$. Standard cross-entropy loss is used on $E_L$; however, to this loss term is added the Kullback-Leibler divergence ($D_{KL}$), or relative entropy, between the unlabeled pairs of augmented and non-augmented examples (hence the term consistency training) as shown in Equation 2[4]. The challenge is performing data augmentation of the unlabeled examples while preserving the expected class label.

[4]Note that the UDA methodology also has tuning parameters such as softmax temperature and confidence threshold masking for the probabilities in the $D_{KL}$ term

$$Loss_{Total} = -log(q(E_L)) + -log\frac{q(E_U)}{q(E'_U)} \quad (2)$$

Two methods for performing this augmentation are discussed in [1], forward-backward translation and TF-IDF word replacement. For the purposes of this analysis only the TF-IDF word replacement augmentation strategy was used. However, in Section V, additional text augmentation methods are considered. The TF-IDF word replacement strategy was chosen because it was judged that people and locations would be essentially keywords that would be useful for the book classification task, and this strategy would preserve those. This data augmentation strategy works by generating a TF-IDF score for each word in a given sequence (prior to tokenization). A probability factor is then used to tune the likelihood of each word being replaced, with words having lower TF-IDF scores (relative to the maximum occuring in the sequence) being more likely to be replaced (See Appendix B to [1] for more details). The entire set of 15,001 available training examples is augmented and made available for using during the UDA training process. From the augmented examples provided in Table 4, it can clearly be seen that expected keywords with high TF-IDF values such as the names **Edd** and **Sansa** are preserved while things with expected low TF-IDF values, such as punctuation is more likely to be replaced.

## III. Analysis Results

Given that the dataset used for the studies discussed herein is novel, baseline accuracy scores were generated and are provided as Figure 1. As discussed in Section ii, the baseline accuracies were generated using a Naive-Bayes approach on the tokenized sequences. During the baseline model evaluation, the simple CountVectorizer model using bigrams showed superior performance as evaluated on the development set and was thusly used for the final evaluations. Figure 1 clearly shows the expected behavior between accuracy and training

**Table 4:** *Examples of TF-IDF Data Augmentation*

| Replacement Probability | Data Type | Text |
|---|---|---|
| **p=0.1** | Original | It was strangely comforting to see Edd's dour face again**.** How goes the restoration work? He **asked** his old steward. |
| | Augmented | It was strangely comforting to see Edd's dour face again **Owen** How goes the restoration work? He **handled** his old steward. |
| **p=0.3** | Original | No**,** she remembered thinking, **not** every face, my lord. No one was smiling now. The **looks** the sparrows gave her were dull, sullen, hostile**.** |
| | Augmented | No **dwell** she remembered thinking, **worship** every face, my lord **machines** No one was smiling now. The **treachery** the sparrows gave **see** were dull, sullen, hostile **rooftops** |
| **p=0.5** | Original | **He will,** Sansa **said,** heart soaring**. Oh,** I **know he** will**. The straw on the floor stank of** urine**. There was** no **window**, no bed, **not even a** slop bucket**.** |
| | Augmented | **Feel falling Quick** Sansa **custody circle** heart soaring. **trickle loans** I **brown shift** will **crowd priests unadorned stone cleaved roofless narrowed dipped** urine **Illyrio fleas owns** no **touch**, no bed, **congratulate smiling And** slop bucket **acolytes** |

**Table 5:** *Classification Error Rates*

| Model | Training Set Size | | | | |
|---|---|---|---|---|---|
| | 20 | 200 | 2000 | 5000 | 12000 |
| **Baseline** | **0.747** | **0.622** | 0.422 | 0.382 | 0.322 |
| **BERT FT** | 0.770 | 0.723 | **0.369** | **0.274** | **0.242** |
| **XLNet FT** | **0.747** | 0.667 | 0.375 | 0.307 | 0.278 |

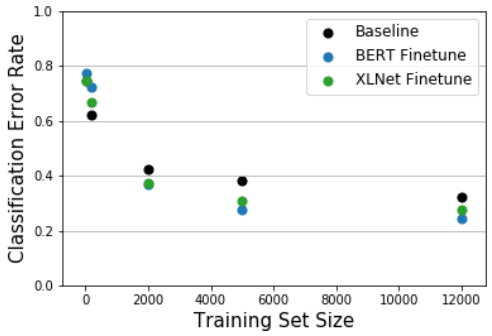


**Figure 1:** *Baseline Error Rates*

set size and motivates the exploration of UDA as a potential means for increasing accuracy when only limited labeled data is available. Error rates are explicitly given in Table 6.

Finetuned model accuracies are also presented in Table 6, one can see the clear benefit of the more advanced models, but only with sufficient training data (i.e., $\geq$ 2000). With fewer data, the baseline actually outperforms the BERT and XLNet models with training set sizes less than 2000. Above this level the BERT model is actually highest performing, which was a somewhat surprising result given that XLNet models have been shown to result in state-of-the-art accuracies for text classification [2]. It should be noted, however, that the XLNet to BERT multi-class text classification comparison provided in [2] was done with only the large models (i.e., with twice as many attention layers). It is unclear whether the base models showed the same trends as the large models. In general, the results met with expectation. With the largest training set size, the multi-class error rate approaches 24%. For the Amazon-5 and Yelp-5 text classification sets, finetuned BERT [3] and XLNet [2] models, 28 and 35% by comparison.

Error rates from the finetuned and optimized semi-supervised models are also presented in Table **??** along with relative and absolute change in the error rate with respect to the fine-

**Table 6:** *Classification Error Rates with UDA*

| Model | Training Set Size | | | | |
|---|---|---|---|---|---|
| | 20 | 200 | 2000 | 5000 | 12000 |
| **Baseline** | **0.747** | **0.622** | 0.422 | 0.382 | 0.322 |
| **BERT FT** | 0.770 | 0.723 | **0.369** | **0.274** | **0.242** |
| **XLNet FT** | **0.747** | 0.667 | 0.375 | 0.307 | 0.278 |
| **BERT UDA** | 0.776 | 0.676 | **0.335** | **0.291** | **0.205** |
| **XLNet UDA** | 0.770 | 0.723 | **0.369** | **0.274** | **0.242** |
| **ACE** | -0.006 | 0.047 | 0.034 | -0.017 | 0.037 |
| **RCE** | -0.8% | 6.5% | 9.2% | -3.0% | 15.3% |

ACE = Absolute Change in Error
RCE = Relative Change in Error

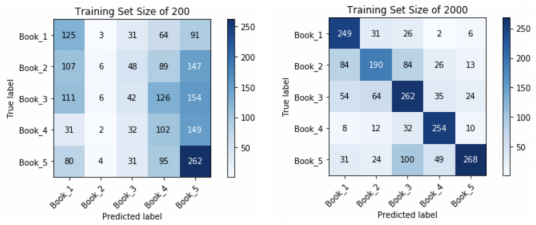


**Figure 2:** *BERT Finetune Confusion Matrices*

tuned models with the same labeled dataset size.

Investigation of the BERT model errors for the training set size of 200 indicated that even though the training sets were balanced, nearly none of the test set examples were being predicted to come from the second novel, *Clash of Kings*. However, as additional training data class prediction becomes much more balanced, with no single class standing out as particularly troublesome as shown in Figure 2.

## IV. Experiments

Several experiments aimed at understanding performance tradeoffs with hyperparameters in both the finetuned and UDA models were executed. For finetuning, these studies were focused primarily on batch size, hidden layer dropout rate and learning rate, and all experiments were only evaluated relative to the development set. While other parameters were also studied, these three provided the largest performance changes. An example suite of these

**Table 7:** *Select XLNet Finetuning Experiments*

| Training Set Size | Batch Size | Dropout Rate | Learning Rate | Dev. Error Rate |
|---|---|---|---|---|
| **12000** | 32 | 0.10 | 3E-5 | 0.250 |
| **12000** | 32 | **0.15** | 3E-5 | 0.303 |
| **12000** | 32 | **0.20** | 3E-5 | 0.353 |
| **12000** | **16** | **0.15** | 3E-5 | 0.389 |
| **12000** | 16 | 0.15 | **5E-5** | 0.377 |

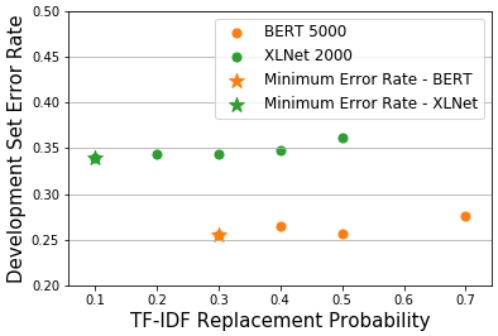


**Figure 3:** *TF-IDF Replacement Probability Experiments*

senstivities, performed for the XLNet finetuned model are shown in Table 7.

For the semi-supervised portion of this analysis a suite of sensitivity evaluations were run for each training set size to optimize the hyperparameters. In addition to those available during finetuning, the UDA process also includes several additional hyperparameters which can make time consuming. Although initial studies were performed following the guidance provided in [1], it was determined that this tuning was quite problem specific. For example, the TF-IDF replacement probability most commonly used in the [1] study was stated to be 0.7; however, the studies presented herein optimized at lower replacement probabilities for both BERT and XLNet models, as shown in Figure 3[5]. Further no benefit was observed from either confidence masking or by decreasing softmax temperature (i.e., accentuating probability differences) on the unsuperivsed data.

[5] All other hyperparameters held constant.

## V. Future Work

The work presented herein only looked at a single data augmentation technique during the UDA application. Future work should evaluate whether the back-translation augmentation methodology applied in [1] provides improved performance over the TF-IDF augmentation. Additionally recent work by Sun, et al., [8] provides exciting new methods for back-translation by generating a distinct translation from each head in a multi-attention head transformer network. The effectiveness of using different TSA schedules for different amounts of training data, seen in [1], was reproduced by this work. Given that these are relatively simple models for how labeled data is introduced during the UDA process, additional work to understand and optimize these schedules should be pursued. In addition to furthering the UDA methodology, further work that looks at both BERT and XLNet models should further simplify the modeling. While either model can be run from the Colaboratory notebook and some elimination of redundant methods was implemented [6], there is still significant streamlining that can be done to enhance simplicity and flexibility for future users of this process.

## VI. Conclusions

## References

[1] Qizh Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong and Quoc V. Le. (2019). Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848v4*, September 2019.

[2] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le (2019). XLNet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237v1*, June 2019.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez., Lukasz Kaiser, and Illia Polosukhin (2017). Attention is all you need. *arXiv arXiv:1706.03762v5*, December 2017.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv arXiv:1810.04805v2*, May 2019.

[5] Mike Schuster and Kaisuke Nakajima (2012). Japanese and Korean Voice Search *2012 IEEE International Conference on Acoustics, Speech, and Signal Processing arXiv:1609.08144*, 2012.

[6] Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv arXiv:1901.02860v3*, June 2019.

[7] Rico Sennrich, Barry Haddow, and Alexandra Birch (2016). Improving neural machine translation models with monolingual data. *arXiv arXiv:1511.06709*, June 2016.

[8] Zewei Sun, Shujian Huang, Hao-Ran Wei, Xin-yu Dai and Jiajun Chen. (2019). Generating Diverse Translation by Manipulating Multi-head Attention. *arXiv arXiv:1911.09333v1*, November 2019.

[6] Project GitHub repository