PRADO: Projection Attention Networks for Document Classification On-Device

Prabhu Kaliamoorthi
Google Research
Mountain View, CA, USA
prabhumk@google.com

Sujith Ravi
Google Research
Mountain View, CA, USA
sravi@google.com

Zornitsa Kozareva
Google
Mountain View, CA, USA
zornitsa@kozareva.com

Abstract

Recently, there has been a great interest in the development of small and accurate neural networks that run entirely on devices such as mobile phones, smart watches and IoT. This enables user privacy, consistent user experience and low latency. Although a wide range of applications have been targeted from wake word detection to short text classification, yet there are no on-device networks for long text classification.

We propose a novel projection attention neural network PRADO that combines trainable projections with attention and convolutions. We evaluate our approach on multiple large document text classification tasks. Our results show the effectiveness of the trainable projection model in finding semantically similar phrases and reaching high performance while maintaining compact size. Using this approach, we train tiny neural networks just 200 Kilobytes in size that improve over prior CNN and LSTM models and achieve near state of the art performance on multiple long document classification tasks. We also apply our model for transfer learning, show its robustness and ability to further improve the performance in limited data scenarios.

1 Introduction

One of the fundamental tasks in Natural Language Processing is related to long text classification. Given a document, the goal is to assign one or more categories of interest to the text. This task is of high importance as it has wide applications in spam detection (Jindal and Liu, 2007), product categorization (Kozareva, 2015), sentiment classification (Pang and Lee, 2008) and it also plays an important role for improving document retrieval and ranking (Deerwester et al., 1990).

For a long time, the most successful text classification approaches relied on sparse lexical features such as n-grams, which are later used by linear or kernel models (Joachims, 1998; McCallum and Nigam, 1998; Joulin et al., 2016). However, with the recent advancements in deep learning, various neural network architectures like CNN (Kim, 2014), LSTM (Zhang et al., 2015), hierarchical attention mechanisms (Yang et al., 2016) showed improvement in performance.

Recently, (Ravi and Kozareva, 2018) and (Ravi and Kozareva, 2019) showed the importance of building on-device neural models for short text classification, which preserve user privacy, enable consistent user experience and most importantly perform inference on the device. One of the biggest challenges is how to fit these large and complex neural networks on devices with limited memory and computation capacity while still maintaining high performance. (Ravi and Kozareva, 2018, 2019) developed on-device self-governing neural networks (SGNN) and (SGNN++) based on locality-sensitive projections (Ravi, 2017, 2019). Those methods were evaluated on short text classification tasks such as dialog act and user intent understanding and outperformed prior RNN work (Khanpour et al., 2016; Ortega and Vu, 2017).

In this work, we take one step further by proposing a novel projection attention neural network called PRADO . Unlike SGNN which has static projections, PRADO combines trainable projections with attention and convolutions allowing it to capture long range dependencies and making it a powerful and flexible approach for long text classification. We study the impact of different hyperparameters on accuracy vs model size. We also address the problem of producing compact architectures by develop a quantized version of PRADO . In a series of experimental evaluations on multiple long text classification tasks, we show that our approach PRADO improves over prior baselines and

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 5012–5021,
Hong Kong, China, November 3–7, 2019.

neural networks such as character and word level CNNs and LSTMs. The main contributions of our work are as follows:

- Novel on-device projection attention neural network PRADO which combines transferable projections with attention and convolution for long text classification.
- Exhaustive experimental evaluation on multiple long text classification tasks, outperforming traditional feature engineered linear classifiers and deep learning approaches like CNN and LSTM.
- Quantized PRADO network, which results in tiny 200 Kilobytes in size model that improves over prior CNN and LSTM models.
- Applicability of PRADO for transfer learning, showed its robustness and ability to further improve performance in limited data scenarios.

2 Related Work

Early work on text classification relied on sparse lexical features such as n-grams and linear classifiers (Joachims, 1998; McCallum and Nigam, 1998; Joulin et al., 2016). But with the raise of deep learning, various CNN and LSTM approaches lead to significant improvement in performance and reaching state-of-the-art results. (Kim, 2014) used CNN architecture from computer vision for text classification. (Johnson and Zhang, 2015), used high-dimensional one hot vector and later introduced character-level CNN that achieved even more competitive results. (Tai et al., 2015) used tree structured LSTM for classification, while (Tang et al., 2015) use CNN or LSTM to capture sentence vector followed by bidirectional gated recurrent network which composes the vectors to get a document vector. Recently, (Yang et al., 2016) introduced hierarchical attention neural networks, which captures document representation by incorporating knowledge of the document structure into the model. This approach reaches the best performance on large set of text classification tasks.

The aforementioned prior work mostly focuses on building the best neural network model independent of any model size or memory constrains. However, recent work by (Ravi and Kozareva, 2018, 2019) show the importance of building on-device text classification models that can preserve user privacy, provide consistent user experience and most importantly are compact in size, while yet achieving state-of-art results. Previously, to build lightweight text classification approaches (Ravi, 2013) proposed fast sampling techniques, while (Bui et al., 2018) incorporated deep neural networks with graph learning. While successful, such approaches resulted in large models for response completion (Pang and Ravi, 2012) and Smart Reply (Kannan et al., 2016).

To address the challenge of fitting huge deep neural network on-device, (Ravi and Kozareva, 2018) developed a novel self-governing neural networks (SGNNs) that learns projections on the fly leading to small models. SGNN was applied on short text classification tasks such as dialog act and user intent understanding and showed significant improvement over state-of-the-art RNN (Khanpour et al., 2016) and RNN with attention (Ortega and Vu, 2017) approaches. In this work, we take one step further by developing trainable projection network with attention mechanism that captures long range dependencies making it a powerful and flexible approach for long text classification. In addition, we address the problem of producing compact architectures for text classification when we have limited amount of memory. FastText (Joulin et al., 2016) proposed product quantization to store word embeddings and have carried out evaluation which show models that require two orders of magnitude less memory. We use quantization techniques and show that we achieve 10x to 100x compression rate while still maintaining high performance and improving upon prior CNN and LSTM work. Unlike prior on-device text classification work, we also apply our model in a transfer learning scenario, which demonstrated the robustness of our approach and ability to further improve performance in limited data scenarios. Next, we describe the technical details of our approach, followed by experimental evaluation and results.

3 PRADO: Projection Attention Network

Figure 1 shows the overall architecture of our proposed network PRADO . It consists of a projected embedding layer, a convolutional and attention encoder mechanism and a final classification layer. We describe each component in detail below and contrast them with existing methods.

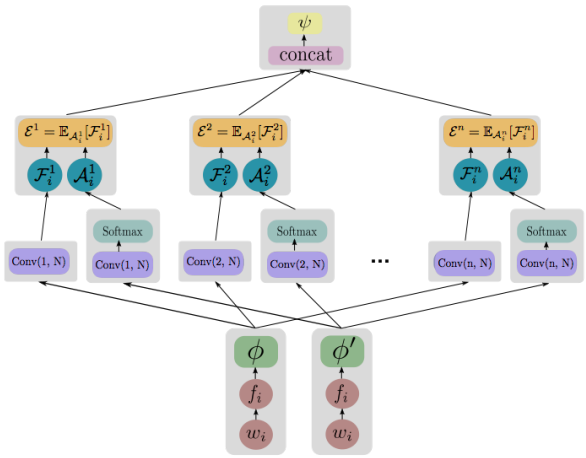


Figure 1: PRADO Model Architecture

3.1 Projected Embedding Layer

Let us assume that the input text has T tokens or words. w_i represents the i-th word, where $i \in \{0, ...T-1\}$. If V is the number of words in the vocabulary, including an out of vocabulary token that represents all missing words, then each word w_i is mapped to $\delta_i \in \mathbb{R}^V$. The first component in most neural networks designed for language tasks uses an embedding layer with trainable parameters $W \in \mathbb{R}^{d \cdot V}$ to map words to fixed length d-dimensional vectors $e_i = W\delta_i$, where $e_i \in \mathbb{R}^d$ are the word vectors that are processed by the rest of the network. A large fraction of the parameters in the network is concentrated in W, since V often has to be large (upto hundreds of thousands or millions of words) to obtain good performance. Furthermore, by choosing V upfront we are assuming that words or phrases relevant for the classification task are known apriori, which may not be true. It should be noted that though we express the operation to obtain the word vector e_i as a matrix multiplication, in reality it is a look-up of the corresponding row in the embedding matrix as δ_i is modeled using the Dirac delta function.

Embeddings via Trainable Projections: Our approach PRADO replaces this embedding with a projection approach to build the word encoder. Instead of mapping w_i to δ_i, we map it to f_i using a projection operator $\mathbb{P}$. Recent work (Ravi, 2017; Ravi and Kozareva, 2018; Ravi, 2019) has shown that projection-based neural approaches can help train compact neural networks that achieve good performance on certain language tasks. These networks learn robust representations (Sankar et al., 2019a) that can be also transferred to other tasks (Sankar et al., 2019b). We follow a similar strategy but unlike the static projections used in these works, we propose a new type of projection that decomposes the operation and makes the projection *trainable*, leading to more powerful encoders capable of capturing contextual information for long-text classification while maintaining a very low memory footprint. Our method does not rely on a fixed vocabulary.

The projection operator we use in this work first fingerprints the words and extracts B bit features from the fingerprint. The word vectors e_i are obtained using a neural network layer $e_i = \phi(f_i)$. This allows us to generate *compact* embeddings, *train the projection encoder* layer better and apply further optimizations like batch normalization (Ioffe and Szegedy, 2015) in this step. First, individual tokens w_i in the input text are fingerprinted using a hashing function to generate $2B$ bits. The projection operator $\mathbb{P}$ then maps every consecutive two-bit sequence to the set $\{-1, 0, 1\}$ resulting in a vector $f_i \in \{-1, 0, 1\}^B$. We chose this specific form since it would enable further optimization such as those described in (Li and Liu, 2016) to reduce the computation in the first layer. We note that there could be alternative modeling choices for the specific form of the projection operator $\mathbb{P}$. Any projection operator that maps bits from the fingerprint to a bounded range is expected to perform equally well. $\phi(.)$ is a trainable function with $B \cdot d$ parameters that maps B-dimensional projection features into d-dimensional word embedding vectors that are computed dynamically during training and inference. In practice, $B \in [128, 512]$ and $d \in [32, 96]$ are tiny compared to V.

3.2 Convolution & Attention over Projections

Next, we introduce an encoder mechanism to map a sequence of projected word embeddings $e_0...e_{T-1}$ to a fixed length vector that represents the entire input text. There are many studies that use 1d convolutions on the word vectors and perform pooling to reduce the sequence to a fixed length vector. But we observe that most words or tokens in a sentence are not relevant for any classification problem, as a result methods like average pooling after convolutions do not effectively reduce the most relevant information needed for the task, especially when the text contains several sen-

tences. Other pooling methods like max or min do not let gradients flow effectively during backpropagation, making it difficult to train the network. To overcome this, we propose a method that uses convolutions and attention mechanism over the word projections and generate a fixed length encoding for the input text.

Projected Attention: In our approach, we use two independent convolutional networks for this step. First one, which we refer to as the *projected feature* network $\mathcal{F}$ captures the features that are useful for the classification task. This is comparable to the convolution networks used in existing studies except we perform convolutions over the sequence of projected word embeddings e_i.

$$\mathcal{F}_i^n = Conv(e_i, n, N) \tag{1}$$

where n is the convolution kernel width, N is the number of output channels in the convolution and $\mathcal{F}_i^n \in \mathbb{R}^N$. The second one, which we refer to as *attention network* $\mathcal{A}$, captures the importance of these features for the task.

$$\mathcal{W}_i^n = Conv(e_i, n, N) \tag{2}$$

We compute softmax over the sequence dimension of the results of $\mathcal{A}$. This provides a distribution over the word sequence that captures the relevance of features at different positions.

$$\mathcal{A}_i^n = \frac{e^{W_i^n}}{\sum_i e^{W_i^n}} \tag{3}$$

We compute an expectation using distribution $\mathcal{A}$ on $\mathcal{F}$ that turns the sequence into a fixed length encoding $\mathcal{E}^n$ for convolution kernel n.

$$\mathcal{E}^n = \sum_i \mathcal{A}_i^n \mathcal{F}_i^n \tag{4}$$

Our pooling scheme reduces to average pooling if the $\mathcal{A}_i^n$ is uniform over the sequence dimension and it becomes max or min pooling if $\mathcal{A}_i^n$ is a Dirac delta in the maximum or minimum value.

3.3 Sequence Convolution Kernels

For the convolution and attention encoder $\mathcal{E}^n$ above, we separately apply n-gram kernels of varying sizes $n = 1, 2, 3, ...$ In addition to n-grams, we used *masked* convolution kernels that simulate the effect of skip-grams. The masking effectively zeros out certain entries in the convolution kernel as shown in Figure 2. Each kernel n generates a corresponding fixed length encoding $\mathcal{E}^n$ of the input sequence.

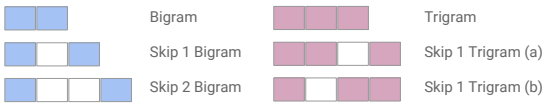


Figure 2: Skip-gram Simulation with Masked Convolution Kernel

3.4 Classification Layer

The convolution kernel width n makes the network react to various word n-grams with a configurable parameter N for each n-gram. We compute various n-gram and skip-gram convolution features and concatenate them to form a fixed length representation for the input document x_k:

$$\text{TextEncoder}(x_k) = \text{concat}(\mathcal{E}^1, \mathcal{E}^2, ...) \tag{5}$$

Finally, we use a feed-forward network with fully-connected layer over the fixed length text encoding for classification.

$$\text{output} = \psi(\text{TextEncoder}(x_k)) \tag{6}$$

We train the network with cross entropy loss and apply softmax over the output layer to obtain predicted probabilities y_k^C for each class C during inference.

4 Experiments & Results

4.1 Data Sets

We evaluate the performance of our approach on large scale document classification tasks, which are widely used in the research community.

- **Yelp** reviews from the Yelp Challenge (Tang et al., 2015) with ratings from 1 to 5.
- **Amazon** reviews from (Zhang et al., 2015) with ratings from 1 to 5.
- **Yahoo Answers** from (Zhang et al., 2015) with documents contain question title, question context and best answer and ten classes such as: Society & Culture; Science & Mathematics; Health; Education & Reference; Computers & Internet; Sports; Business & Finance; Entertainment & Musical; Family & Relationship; Politics & Government;

Table 1 shows the characteristics of each data set. We use the same test sets as (Tang et al., 2015).

Data Sets	#Classes	Training	Test
Yelp	5	650K	50K
Amazon	5	3M	650K
Yahoo Answers	10	1.4M	60K

Table 1: Data Set Characteristics

Parameter	Description
B	Dimension of the projection
d	Dimension of the word embedding computed on the fly
N_1	Number of unigram convolution channels
N_2	Number of bigram convolution channels
N_3	Number of trigram convolution channels
N_4	Number of 4gram convolution channels
N_5	Number of 5gram convolution channels
S_2^1	Number of skip 1 bigram convolution channels
S_2^2	Number of skip 2 bigram convolution channels

Table 2: Hyper Parameters Searched

4.2 Experimental Setting

We setup our experimental evaluation, as follows: given a long text classification task and a data set, we construct a model with the hyper-parameters listed in Table 2 and we use a hyper-parameter search technique to find the optimal model for each data set. In addition to the parameters listed in Table 2, the search method also looks for optimal learning rate schedule and regularization scale. For the purpose of hyper-parameter search, we set aside 8% of the training data and use it as development set. The search method optimizes the *Accuracy* on the development set. Once found, we use the optimal model to evaluate *Accuracy* on the test set.

4.3 Implementation Details

Unlike prior document classification neural networks (Zhang et al., 2015; Tang et al., 2015; Yang et al., 2016) which rely on pre-trained word embeddings, our approach PRADO learns the projection weights on the fly during training (i.e word embeddings (or vocabularies) do not need to be stored). Prior to learning the projections, we did a simple pre-possessing that normalized the text to lowercase, introduced blank space before and after punctuation to make sure they are treated as separate tokens and tokenized the text by space.

We used different regularization scales for the initial fully connected layer and the rest of the network, as the majority of the parameters were in the first layer that computes the word embedding vectors on-the-fly. We used Adam optimizer (Kingma and Ba, 2014) with exponential learning rate schedule. For regularization, we used dropouts after the first layer and also distorted the input text by randomly inserting, deleting and transposing characters in the token with small probability.

4.4 Results and Model Comparisons

It is important to recall that the main goal of our work is to develop fast and efficient on-device neural text classification approach, which can achieve near state-of-the-art performance while satisfying the on-device small size and memory resource constrains. Therefore, it is not fair to directly compare PRADO on-device performance against existing approaches which do inference on cloud without constraints. Yet, we compare our approach against well established baselines and prior non-on-device work taking into consideration these differences. Table 3 shows the obtained results for each data set and method.

Baseline Comparison: We use the same baselines as described in (Zhang et al., 2015; Tang et al., 2015). They are traditional approaches, which rely on hand-crafted features such as bag-of-words with TFIDF and n-grams with TFIDF, and use linear or kernel classifiers. As it can be seen in Table 3, our PRADO approach consistently outperforms all baseline methods with +4.8 to +12.2 for Yelp, +5.9 to +17 for Amazon and +1.3 to +11.8 for Yahoo data sets. This definitely shows the power of the trainable projection on-device neural networks and attention mechanism.

On-device Comparison: We also show comparison against prior on-device neural network approach (Ravi and Kozareva, 2018). Their SGNN approach was targeted towards short text classification tasks and as shown in Table 3, our PRADO model achieves upto +40% improvement over SGNN demonstrating that PRADO is more powerful and suited for long text classification.

Deep Learning Comparison: Similarly, we also compare our PRADO approach against recent neural networks such as LSTM (Zhang et al., 2015), character and word-based CNNs (Zhang et al., 2015; Tang et al., 2015), Convolutional GRNN (Tang et al., 2015) and hierarchical attention (Yang et al., 2016). We compare performance for all data sets against (Tang et al., 2015; Zhang et al., 2015) since their test data sets are exactly the same as ours. However, (Yang et al., 2016) uses different test data sizes for the Yelp and Ya-

Data Set	Yelp	Amazon	Yahoo
PRADO	**64.7**	**61.2**	**72.3**
PRADO 8-bit Quantized	**65.9**	**61.9**	**72.5**
SGNN (Ravi and Kozareva, 2018)	35.4	39.1	36.6
HN-ATT* (Yang et al., 2016)	-	**63.6**	-
HN-MAX* (Yang et al., 2016)	-	**62.9**	-
HN-AVE* (Yang et al., 2016)	-	**62.9**	-
LSTM-GRNN (Tang et al., 2015)	**67.6**	-	-
Conv-GRNN (Tang et al., 2015)	**66.0**	-	-
CNN-char (Zhang et al., 2015)	62.0	59.6	71.2
CNN-word (Tang et al., 2015)	61.5	-	-
CNN-word (Zhang et al., 2015)	60.5	57.6	71.2
Paragraph Vector (Tang et al., 2015)	60.5	-	-
LSTM (Zhang et al., 2015)	58.2	59.4	70.8
SVM + Bigrams (Tang et al., 2015)	62.4	-	-
SVM + Unigrams (Tang et al., 2015)	61.1	-	-
SVM + AverageSG (Tang et al., 2015)	56.8	-	-
SVM + SSWE (Tang et al., 2015)	55.4	-	-
BoW TFIDF (Zhang et al., 2015)	59.9	55.3	71.0
ngrams TFIDF (Zhang et al., 2015)	54.8	52.4	68.5

Table 3: Evaluation Results

hoo Answers evaluation, therefore we do not report their results here. Another interesting aspect is that unlike prior work (Zhang et al., 2015; Tang et al., 2015; Yang et al., 2016) which rely on large word embeddings (with pre-training in many cases), our approach computes dynamically the projection embedding vectors. As shown in Table 3, PRADO significantly outperforms existing neural networks approaches like LSTM, CNN-char and CNN-word with +1.1 up to +6.5% depending on the task and data set, and it achieves comparable results to the hierarchical attention models of (Yang et al., 2016). This is very impressive given that PRADO produces magnitudes smaller and compact neural networks.

4.5 Impact of Hyper-parameters on Performance

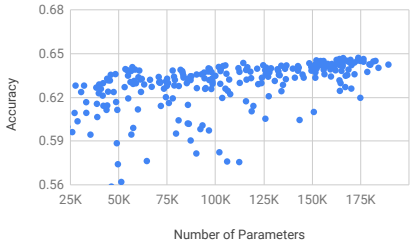
Figure 3 shows *Accuracy* vs Model Size as a result of the different hyper-parameters explored during search. The figure shows results for each data set individually. Overall, PRADO's *Accuracy* improves as the number of parameters increases.

4.6 Model Size: PRADO vs Smaller RNNs

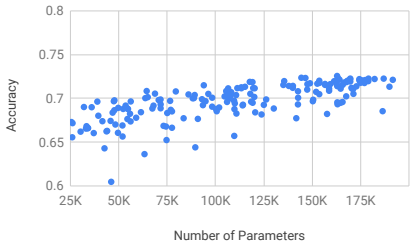
We further compare our PRADO model against smaller-sized variants of widely-used recurrent (LSTM) models. This study helps analyze the effectiveness of PRADO compared to other small neural models and answer the question: *Can popular RNN models be shrunk down to the same size as PRADO and still achieve high performance?* To construct baseline neural models at smaller sizes, we use an LSTM architecture with 64 hidden units as the state size and vary the input vocabulary size (i.e., picking top K words ordered by frequency) and embedding dimensions d. Table 4 compares the performance of PRADO with different small and medium-sized LSTM models (achieved by varying K, d). Our results show that PRADO achieves the best performance with the lowest footprint (just 175K parameters) and high compression ratios (up to 100x smaller) compared to standard LSTM models.

	Compression (PRADO)	#Parameters	Accuracy
LSTM	100x	18.17M	60.4
	43x	7.6M	60.3
	11x	1.9M	60.2
	3x	504.7K	59.9
	1x	179.5K	59.1
PRADO	**1x**	175K	**64.7**

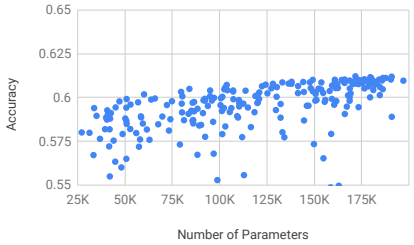
Table 4: Model Size vs. Performance for PRADO and small LSTMs on Yelp. Compression ratio for PRADO is shown wrt corresponding LSTM models.



(a) Yelp 5



(b) Yahoo Answers 10



(c) Amazon 5

Figure 3: Model Size (*number of parameters* on x-axis) vs. Performance (*accuracy* on y-axis).

4.7 PRADO Analysis and Discussion

N-gram Attention Focus: To better understand our model, we analyze PRADO's attention distribution. We used a trained PRADO model from a particular data set and computed the attention distribution of the projected n-gram features for all samples in the test set. Then, we pull the word or word sequence index with the maximum attention and order them by frequency of the class. Table 5 shows examples of the bigrams for 5 and 1 star Yelp reviews. Note, that we did not select the vocabulary for the data, the model automatically learned to associate the words with the classes based on the trained projections.

Skipgrams Attention Focus: Similarly, we conduct an analysis for the attention distribution of the skip-1-bigrams channel. Table 5 shows sample of the most frequent entries for the 5 and 1 star Yelp reviews. The model captures and learns basic regular expressions. For instance, the skip gram "*waste * time*" captures " *waste your time*" or "*waste of time*". Similarly "*worst * ever*" captures "*worst food ever*", "*worst service ever*". Our analysis shows that overall our trainable projection with attention and convolution learns embedding representations that are powerful and capture the semantic similarity of words and phrases. This information helps PRADO during classification.

Bigrams	
5 Star	1 Star
highly recommend	*zero stars*
love this	*horrible customer*
hands down	*1 star*
was perfect	*disgusting and*
top notch	*no stars*
amazing service	*better off*
Skip-1-Bigrams	
5 Star	1 Star
*waste * time*	*worst * ever*
*was * reasonable*	*give * stars*
*felt * comfortable*	*a * star*
*is * delicious*	*waste * money*
*you * comfortable*	*horrible * service*
*and * delicious*	*worst * experience*

Table 5: Prado Attention Focus on Yelp Data

5 PRADO Runtime Performance

Our PRADO approach produces compact neural networks with tiny memory footprint. Next, we also show how to further optimize PRADO and help speed up on-device inference during runtime.

5.1 Training with quantization

We train a PRADO model variant with 8-bit quantization as described in (Jacob et al., 2018). This procedure simulates the quantization process during training by nudging the weights and activations towards a grid of discrete levels (2^N levels, where N=8 is the number of bits). We estimate the activation ranges for each training batch and use exponential moving average to smooth the quantization ranges across training steps. (Jacob et al., 2018) noted that by training with quantization, they reached similar accuracy with 8-bit models as floating point ones on several image classification and object detection data sets. For text classification, we observed that training with quantization significantly improves accuracy as shown in Table 3. We believe that this is due to the improved regularization as quantization has the highest impact on Yelp. This dataset has relatively few training samples per class (see Table 1) which causes the model to overfit the training data and regularization provided by the operation that sim-

ulates quantization during training helps it generalize better. Furthermore, the model size of 8-bit quantized PRADO models is equal to the number of parameters. Figure 3 shows that PRADO can reach the performance reported in the Table 3 with model size of less than 200 Kilobytes. PRADO starts getting competitive results on the same data sets with tiny model size as low as 25 Kilobytes.

5.2 Computational Cost for Inference

We evaluate the computational cost of PRADO models for inference wrt *floating point (or integer) operations* and *latency* (in milliseconds). The number of floating point or integer operations in our model is dominated by the below factor

$$Td\left(2B + \sum_{k=1}^{5} kN_k + \sum_{s=1}^{2}(s+2)S_2^s\right) \quad (7)$$

which includes the operations from the first fully connected layer and the convolutional layers. It can be seen that our method scales linearly with the number of time steps T and the dimension of the projected word embedding d. We measured the latency of processing a document with 1000 words using our quantized PRADO model on a Nexus 5x mobile phone to be between 20 to 40 ms.

6 Transfer Learning with PRADO

Recent popularity of several pre-trained word embedding approaches can be primarily attributed to their success and effectiveness at transfer learning for language tasks. Model representations trained on a data-rich domain can be leveraged for tasks and domains in limited-data scenarios. However, as we discussed earlier, these methods require storing and looking up huge pre-trained embedding tables unlike our PRADO approach which results in compact models.

Next, we evaluate the effectiveness of PRADO at learning robust representations and extend it for transfer learning scenarios. To establish a baseline, we took 10% of Yelp training data and trained and evaluated a baseline PRADO model with random initialization of weights. We compared this to initializing the parameters from a PRADO model trained on the larger Amazon data set. We ran two different experiments with this initialization.

- **Experiment A: Full Transfer** We freeze all weights in PRADO except the last fully connected classification layer. With this setup there were around 750 trainable parameters in the model.

- **Experiment B: Projection Transfer** We freeze the weights for only the first (projected embedding) layer and allow the convolution layers and the last classification layer to be trainable.

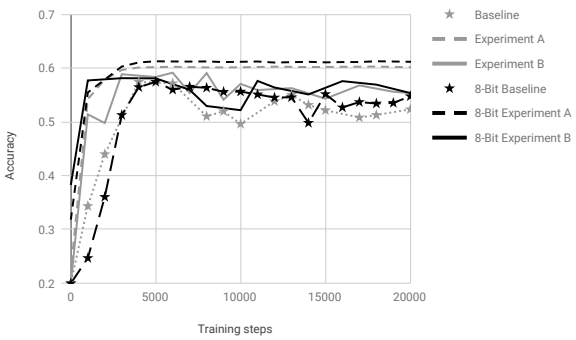


Figure 4: Transfer Learning Results on Yelp Data

We trained the baseline and transfer-learned variants with and without quantization. Figure 4 shows results of the transfer learning runs. We observe that with random initialization, the baseline PRADO model reaches a peak performance of around 57% and starts overfitting on the small Yelp training data set both with and without quantization. When transferring just the *projection layer* and fine-tuning the convolution and classification layers, training converges quickly and it reaches better peak accuracy. This demonstrates that the PRADO projection embeddings trained in one domain, even though tiny in size, captures useful information that can be leveraged to improve classification in another domain with limited training data. This is further improved when transferring the *full* PRADO model and fine-tuning just the last layer. In this case (Experiment A), the model converges to 61% and 60% accuracy with and without quantization respectively. The training also converges quicker than the baseline and it does not overfit anymore. We note that using our approach, a PRADO model transfer-learned on just 10% of training data achieves very competitive performance resulting in less than 10% drop in relative accuracy on Yelp data set (see Table 3).

7 Conclusion

In this paper, we proposed a novel trainable projection on-device neural network with attention,

which is capable of capturing long range dependencies making it flexible for solving long text classification tasks. We introduced trainable projection technique, which via the visualization of the attention mechanism shows that it effectively captures the semantic representation of the document, while still saving on storage and producing compact models. We demonstrated the effectiveness of our approach PRADO by conducting experiments on multiple large scale document classification tasks. The obtained results show that our approach improved upon traditional linear classifiers from 2% to 12%, character and word level CNNs and LSTMs neural approaches with 1% to 6% depending on the data set, task and approach. This is very impressive given the small compact model produced by PRADO . Similarly, the quantized version of our approach had consistent performance. Finally, we applied our model in a transfer learning scenario, which demonstrated the robustness of our approach and ability to further improve performance in limited data scenarios. In the future, we want to extend this approach to more natural language tasks and languages.

References

Thang D. Bui, Sujith Ravi, and Vivek Ramavajjala. 2018. Neural graph learning: Training neural networks using graphs. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 64–71. ACM.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2704–2713.

Nitin Jindal and Bing Liu. 2007. Review spam detection. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 1189–1190.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML'98, pages 137–142.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 103–112. Association for Computational Linguistics.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759.

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, László Lukács, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 955–964.

Hamed Khanpour, Nishitha Guntakandla, and Rodney Nielsen. 2016. Dialogue act classification in domain-independent conversations using a deep recurrent neural network. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2012–2021. The COLING 2016 Organizing Committee.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Zornitsa Kozareva. 2015. Everyone likes shopping! multi-class product categorization for e-commerce. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1329–1333. Association for Computational Linguistics.

Fengfu Li and Bin Liu. 2016. Ternary weight networks. *CoRR*, abs/1605.04711.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive Bayes text classification. In *Learning for Text Categorization: Papers from the 1998 AAAI Workshop*, pages 41–48.

Daniel Ortega and Ngoc Thang Vu. 2017. Neural-based context representation learning for dialog act classification. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages

247–252. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

Bo Pang and Sujith Ravi. 2012. Revisiting the predictability of language: Response completion in social media. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 1489–1499.

Sujith Ravi. 2013. Scalable decipherment for machine translation via hash sampling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 362–371, Sofia, Bulgaria. Association for Computational Linguistics.

Sujith Ravi. 2017. Projectionnet: Learning efficient on-device deep networks using neural projections. *CoRR*, abs/1708.00630.

Sujith Ravi. 2019. Efficient on-device models using neural projections. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5370–5379.

Sujith Ravi and Zornitsa Kozareva. 2018. Self-governing neural networks for on-device short text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 804–810. Association for Computational Linguistics.

Sujith Ravi and Zornitsa Kozareva. 2019. On-device structured and context partitioned projection networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3784–3793.

Chinnadhurai Sankar, Sujith Ravi, and Zornitsa Kozareva. 2019a. On the robustness of projection neural networks for efficient text representation: An empirical study. *ArXiv*, abs/1908.05763.

Chinnadhurai Sankar, Sujith Ravi, and Zornitsa Kozareva. 2019b. Transferable neural projection representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3355–3360.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566. Association for Computational Linguistics.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, pages 1422–1432. The Association for Computational Linguistics.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical attention networks for document classification. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1480–1489.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.