Transformer to CNN: Improved Text Classification for Edge Devices

Yew Ken Chia
Red Dragon AI
Singapore
ken@reddragon.ai

Sam Witteveen
Red Dragon AI
Singapore
sam@reddragon.ai

Martin Andrews
Red Dragon AI
Singapore
martin@reddragon.ai

Abstract

As Deep Learning and NLP models advance they also become more complicated and computationally heavy. This limits the ability of developers to use these models at the edge on phones and low power devices. In this paper, we introduce a new CNN architecture which can be trained by a distillation process from a large-scale model such as OpenAI's Transformer architecture. This student model is then small enough and fast enough to be run on phones. The model can then achieve $300\times$ inference speedup and $39\times$ reduction in parameter count and in some cases, the student model's performance surpasses its teacher on the studied tasks.

1 Introduction

The last year has seen several major advances in NLP modelling, stemming from previous innovations in embeddings [1] [2] and attention models [3] that allow Language Models (LMs) to be trained on very large corpuses : For instance ELMo [4], OpenAI Transformer [5] and recently BERT [6]. In addition, the power of building on LM-enhanced contextualised embeddings, using a fine-tuning approach on task-specific unlabelled data [7], has shown huge benefits for downstream tasks (such as text classification) - especially in a typical industrial setting where labelled data is scarce.

In order to make use of these advances for phones and devices, this work shows how a model distillation process [8] can be used to train a novel 'student' CNN structure from a much larger 'teacher' Language Model. The teacher model can be fine-tuned on the specific task at hand, using both unlabelled data, and the (small number of) labelled training examples available. The student network can then be trained using both labelled and unlabelled data, in a process akin to pseudo-labelling [9] [10]. Our results show it is possible to achieve similar performance to (and surpass in some cases) large attention-based models with a novel, highly efficient student model with only convolutional layers. This reduced model is then able to be used efficiently on mobile devices for a variety of NLP tasks.

2 Model distillation

In this work, we used the OpenAI Transformer [5] model as the 'teacher' in a model-distillation setting, with a variety of different 'student' networks (see Figure 1).

The OpenAI Transformer model consists of a Byte-Pair Encoded subword embedding layer followed by 12-layers of "decoder-only transformer with masked self-attention heads" [3], pretrained on the standard language modelling objective on a corpus of 7000 books. This LM's final layer outputs were then coupled with classification modules and the entire model was discriminatively fine-tuned with an auxiliary language modelling objective, achieving excellent performance on various NLP tasks.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

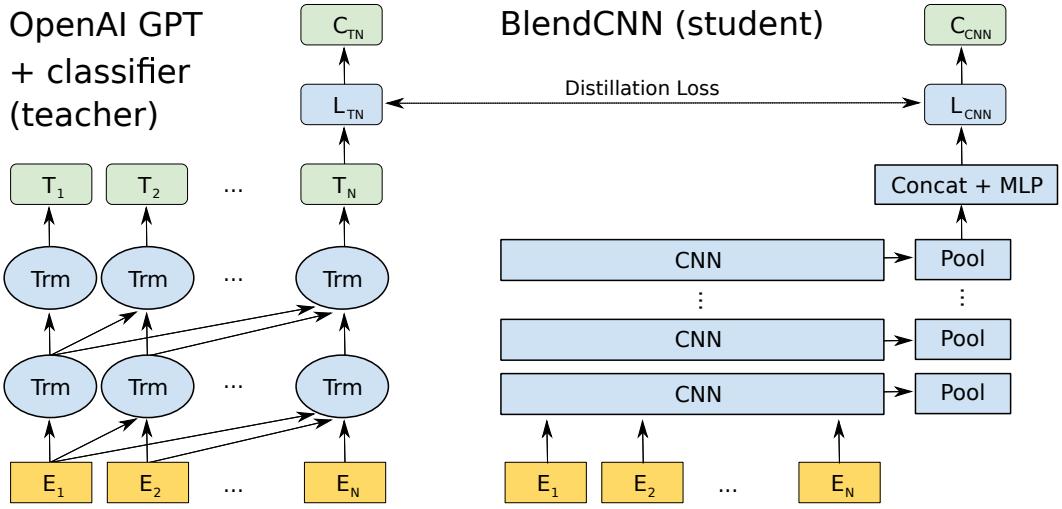

Figure 1: Model architecture with distillation across logits

To optimize for speed and memory constraints of industrial deployment, a variety of different models were trained (a) on the classification task directly; and (b) via distillation [8] of the logit layer output by the pretrained OpenAI classification model.

To combat label-scarcity and improve distillation quality, we inferred distillation logits for unlabelled samples in a pseudo-labelling manner [9] [10], while using transfer learning through pretrained GloVe embeddings [2].

Student models

A number of common network structures were tested in the student role, specifically:

- a two-layer BiLSTM network [11]
- a wide-but shallow CNN network [12]
- a novel CNN structure, dubbed here 'BlendCNN'

The BlendCNN architecture was inspired by the ELMo 'something from every layer' paradigm, and aims to be capable of leveraging hierarchical representations for text classification [13].

The BlendCNN model is illustrated in Figure 1, and comprises a number of CNN layers (with `n_channels`=100, `kernel_width`=5, `activation`=`relu`), each of which exposes a global pooling output as a 'branch'. These branches are then concatenated together and "blended" through a dense network (`width`=100), followed by the usual classification logits layer.

Table 1: Parameter counts and inference timing

	Total parameters[2]	Mobile footprint[3]	Sentences per second[4]
2-Layer BiLSTM[1]	2,406,114	9.4 / 9.4	173.01
KimCNN	**2,124,824**	**7.9 / 8.2**	3154.57
OpenAI Transformer	116,534,790	n/a / n/a	11.76
8-layer BlendCNN	3,617,426	9.4 / 14.1	2392.34
3-layer BlendCNN	2,975,236	8.7 / 11.6	**3676.47**

1 BiLSTM and KimCNN model scores were lower
2 Parameter count estimate using a `vocab_size` of 20000 words, except for OpenAI Transformer (which uses a byte-pair encoding embedding)
3 TensorFlowLite executable size in MB, with/without quantisation
4 Timing measurement used `n_samples`=1000, `batch_size`=32, based on actual time taken for K-80 GPU implementations

Table 2: Scores for standard datasets

	AG News		DBpedia		Yahoo Answers	
TRAINED ON 100 LABELLED EXAMPLES PER CLASS						
TFIDF + SVM	81.9		94.1		54.5	
fastText	75.2		91.0		44.9	
8-Layer BlendCNN	87.6		94.6		58.3	
OpenAI Transformer	88.7		97.5		70.4	
TRAINED BY DISTILLATION[1] OF OPENAI TRANSFORMER						
2-Layer BiLSTM	**91.2**		97.0		70.5	
KimCNN	90.9		97.6		70.4	
3-Layer BlendCNN	**91.2**	/ 88.4[2]	98.2	/ 95.5	**71.0**	/ 63.4
8-Layer BlendCNN	**91.2**	/ 89.9	**98.5**	/ 96.0	70.8	/ 63.4

[1] Distillation training used 100 labelled examples per class, plus 10 times as many unlabelled examples as pseudo-labelled by the OpenAI LM
[2] Smaller entries are results where only labelled examples used
* All CNNs use 100-dimensional trainable GloVe embeddings as input
* Adam optimisation was used, with a constant learning rate of 10^{-3}

3 Experiments

Each of the models was trained and tested on the 3 standard datasets described in [14] : AG News, DBpedia and Yahoo Answers. The experiments had two phases, the first being to evaluate the two baseline methods (TFIDF+SVM [15] and fastText [16]) along with the student network (without the benefit of a LM teacher), and the large LM, with a classification 'head' trained on the task.

The second phase used the large LM in a 'teacher' role, to train the other networks as students via distillation of the LM classifier logits layer (with a Mean Absolute Error loss function).

4 Results

Referring to Table 2, the 3-Layer and 8-Layer variants of the proposed BlendCNN architecture achieve the top scores across all studied datasets. However, the performance of the proposed architecture is lower without the 'guidance' of the teacher teacher logits during training, implying the marked improvement is due to distillation. The additional results given for BlendCNN quantifies the advantage of adding unlabelled data into the distillation phase of the student model training.

Notably from Table 1, the 3-Layer BlendCNN student has $39\times$ fewer parameters and performs inference $300\times$ faster than the OpenAI Transformer which it empirically out-scores.

5 Discussion

For text classifications, mastery may require both high-level concepts gleaned from language understanding and fine-grained textual features such as key phrases. Similar to the larval-adult form analogy made in [8], high-capacity models with task-agnostic pre-training may be well-suited for task mastery on small datasets (which are common in industry). On the other hand, convolutional student architectures may be more ideal for practical applications on phones and devices by taking advantage of parallel computation and a significantly reduced memory and size footprints.

Our results suggest that the proposed BlendCNN architecture can efficiently achieve higher scores on text classification tasks due to the direct leveraging of hierarchical representations, which are learnable (even in a label-sparse setting) from a strong teaching model. This allows for tasks that previously could have only been done in the cloud, to now be done at the edge.

Further development of specialized edge-friendly student architectures could similarly surpass teacher performance if appropriately designed to leverage the knowledge gained from a pretrained, task-agnostic teacher model whilst optimizing for task-specific constraints. and a significantly reduced memory footprint.

References

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[2] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[4] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[5] Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[7] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 328–339, 2018.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[9] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

[10] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. Semi-supervised sequence modeling with cross-view training. *arXiv preprint arXiv:1809.08370*, 2018.

[11] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2873–2879. AAAI Press, 2016.

[12] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[13] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[14] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. Joint embedding of words and labels for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2321–2331. Association for Computational Linguistics, 2018. URL `http://aclweb.org/anthology/P18-1216`.

[15] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[16] Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL*, pages 3–7, 2017.