T-VSE: Transformer-Based Visual Semantic Embedding

Muhammet Bastan
Amazon
Palo Alto, CA, USA
mbastan@amazon.com

Arnau Ramisa
Amazon
Palo Alto, CA, USA
aramisay@amazon.com

Mehmet Tek
Google
Mountain View, CA, USA
mtek@google.com

Abstract

Transformer models have recently achieved impressive performance on NLP tasks, owing to new algorithms for self-supervised pre-training on very large text corpora. In contrast, recent literature suggests that simple average word models outperform more complicated language models, e.g., RNNs and Transformers, on cross-modal image/text search tasks on standard benchmarks, like MS COCO. In this paper, we show that dataset scale and training strategy are critical and demonstrate that transformer-based cross-modal embeddings outperform word average and RNN-based embeddings by a large margin, when trained on a large dataset of e-commerce product image-title pairs.

Figure 1. E-commerce sites have huge amounts of image-text pairs which can be used to learn cross-modal representations. The image is from amazon.com.

1. Introduction

Cross-modal representation learning can leverage the huge amounts of multi-modal image-text data (Fig. 1) that is readily available on e-commerce sites. Each product has an image, a title as a brief description of the product and, often, additional complementary text metadata. Cross-modal learning can utilize this multi-modal data to learn a good representation of the product, which can be used for cross-modal (text-to-image, image-to-text) product search, clustering, de-duplication, recommendation, etc.

Visual semantic embedding (VSE) [3] uses (image, text) pairs to learn a low-dimensional common embedding space (Fig. 2). VSE models typically consist of two-stream neural networks (NN), one CNN branch to encode images and one NN branch to encode the text (Fig. 2). The text consists of a sequence of tokens and requires sequence modeling. Several different models have been employed for text encoding in the literature: RNNs (LSTM/GRU), variants of word2vec, simple word averaging, and more recently **transformers**.

Transformer models [12, 2, 13] have recently achieved state-of-the-art performance on NLP tasks[1] and replaced the previously popular RNN models (LSTM/GRU). This success is due mostly to the self-supervised pre-training of the transformer models on very large text datasets and then fine-tuning on target tasks.

In contrast to their success in text encoding for NLP tasks, transformers have not been shown to work well for cross-modal vision-language tasks, such as VSE. In fact, a recent work [1] reported results claiming "an average embedding language model outperforms an LSTM on retrieval-style tasks; state-of-the-art representations such as BERT perform relatively poorly on vision-language tasks."

In this paper, contrary to [1], we show that transformer-based VSE (T-VSE) actually works much better than word average and RNN-based VSE models. The key to the success of T-VSE is properly training it on a **large** dataset, whereas the standard VSE datasets are relatively small, e.g., MS COCO has 128K (image, caption) pairs. To this end, we constructed a large dataset of 12.1M (image, title) pairs from fashion items listed on amazon.com.

[1] https://gluebenchmark.com

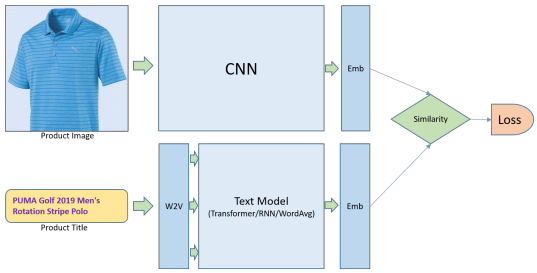

Figure 2. Visual semantic embedding (VSE) learning framework used in this paper.

2. Visual Semantic Embedding (VSE)

We use the classical VSE framework, shown in Fig. 2 for cross-modal retrieval (text-to-image and image-to-text). Given pairs of images and their text descriptions (i_k, t_k), VSE learns embedding functions, or encoders, $f(i_k)$ and $g(t_k)$, by maximizing the similarity $s(i_k, t_k)$ between the positive (image, text) pairs, while minimizing the similarity between negative pairs (i_k, t_j), $k \neq j$. The encoders f and g are typically neural networks (CNNs for images, and RNNs, MLPs or transformers for text), and the similarity function s is the cosine similarity. The VSE model can be trained by optimizing a suitable metric learning loss function, such as the contrastive or the triplet loss.

2.1. Loss Function

In [3], the authors proposed the *Max of Hinges (MH)* loss function, which has proved to be very effective in training VSE models. MH loss is basically a symmetric triplet loss with hard negative mining. Given a batch of N (image, text) pairs (i_k, t_k), image and text encoders f and g, and a similarity function s (inner product), the loss function is defined as two symmetric terms, one for image-to-text and another for text-to-image:

$$L = \sum_{k \neq j}^{N} [\max_j s(f(i_k), g(t_j)) - s(f(i_k), g(t_k)) + m]_+ + \sum_{k \neq j}^{N} [\max_j s(f(i_j), g(t_k)) - s(f(i_k), g(t_k)) + m]_+ \quad (1)$$

where m is a margin value, and $[x]_+ \equiv max(x, 0)$. Hard negative mining is used in both terms, which can be problematic with the presence of duplicates or near-duplicates, as they will be treated as hard negatives. However, the probability of duplicates falling in the same batch decreases as the dataset size increases, and it is also possible to mitigate the problem during sampling and by re-weighting the loss.

2.2. Image and Text Encoders

As image encoder f, we used DenseNet 169, replacing the final classification layer with a linear embedding layer of dimensionality $D = 256$. Higher D gives slightly better accuracy, at the expense of higher computational cost.

Our main focus in this paper is transformer-based VSE models, i.e., the text encoder g uses a transformer model. We also experimented with two other widely used text models for comparison, as in [1]: Word Average and RNN. All three models use the same word embedding layer (W2V in Fig. 2), that projects one-hot encoded text token vectors to word embeddings of low-dimensionality, which in turn are fed to the text model. Finally, a linear embedding layer of size $D = 256$ computes the embedding for the whole text sequence (Fig. 2).

Word Average Model (AVG-VSE). This is the simplest model, taking the word embeddings as input and computing their average as the output. Hence, this model discards the positional information of the input tokens. We inserted an additional fully connected layer of output size 512 before the final embedding layer, as in [1].

RNN Model (RNN-VSE). RNN models have been mainstream in VSE [3]. LSTM and GRU are widely used to capture the sequential nature of text. We used a two-layer, unidirectional GRU, and fed the output of the last hidden state to the final embedding layer. We also experimented with a bidirectional GRU, and with taking the mean of the hidden states, but did not observe any significant difference in performance. A major drawback of RNN models is that they process the input sequentially and are, therefore, not parallelizable.

Transformer Model (T-VSE). Although they were first proposed in [12], transformer models gained popularity with the BERT model [2] which, with the help of pre-training on large unlabeled text data and fine-tuning on target tasks, achieved state-of-the-art results on NLP benchmarks. Many variants of BERT have since been proposed, with slight modifications in the architecture and/or the pre-training algorithm. XLNet [13], RoBERTa [8], ALBERT [7] are only few of them.

Transformers are large –but parallelizable– feed-forward networks, that include self-attention (inner products and softmax), linear, and normalization layers. They can learn context very well due to the self-attention mechanism, and can include a positional encoding layer, whose output is added to the word embeddings and fed to the transformer layers to take into account the ordering of the words. Because of their size, self-supervised pre-training on large unlabeled datasets is key to the success of transformers.

In this paper, we used the DistilBERT model [10], a lightweight BERT model with 6 transformer layers, which is 40% smaller and 60% faster than the original BERT-base model. The DistilBERT model in [10] was trained

by knowledge distillation with BERT-base as the teacher network, achieving 97% of its performance. We only use the network architecture and train it from scratch as described below, without knowledge distillation. We also experimented with a 12-layer DistilBERT model, which has almost the same size as the original BERT base model.

Transformer models typically use a maximum sequence length of 512 for NLP tasks, which is too long for our task of product image-title embeddings. Since the average title length is 17 ± 5, we follow the *three-sigma* rule and set a maximum sequence length of 32, which leads to considerable memory and computation savings.

2.3. Dataset

We constructed a new large scale dataset, **Amazon Fashion 12M** (AF12M)[2], consisting of about 12.1M (product image, product title) pairs from amazon.com in US (Fig. 1). The (image, title) pairs are readily available and no annotation is required.

The dataset was split as follows: 11M (image, title) pairs for training, 100K for validation, and 1M for testing.

2.4. Text Preprocessing and Tokenization

To prepare the product titles for the language model, we first applied Unicode NFKC normalization, followed by ASCII encoding. Next, we removed special characters, corrected common typos, and tokenized the titles according to a vocabulary built from the training set.

The tokenization algorithm and vocabulary size have a significant effect on the network size. For example, [9] used a very large vocabulary (500K), consisting of word unigrams, bigrams, character trigrams and out-of-vocabulary bins, which required large computational resources to train even simple text models. On the other hand, recent state-of-the-art NLP models employ either word-piece or byte-pair encoding [5, 11] tokenization algorithms, resulting in much smaller vocabularies (30K, 50K) [8, 10] and, in turn, smaller and more efficient networks. These tokenization algorithms consider text as a sequence of bytes and are language agnostic. They keep the frequent words as they are while representing rare words with sub-word units, which solves the out-of-vocabulary problem. Based on these insights, we decided to use word-piece tokenization.

The pre-trained transformer models[3] come with vocabularies trained on generic text, but because of the highly specialized nature of the text in our dataset, we found that using vocabularies learned from our training set leads to better results. We used the SentencePiece[4] tokenizer's [6] 'unigram' model [5], which is actually a sub-word tokenizer. We also tried the BPE [11] tokenizer, but it generated less meaningful and more sparse vocabularies, which negatively affects model performance.

We trained vocabularies of size 10K, 20K, 30K and 40K. Larger vocabulary sizes translate to slightly higher accuracy, in exchange for an increased memory and computational cost. A vocabulary of 30K is a good trade-off.

2.5. Training

We trained all three VSE models (AVG-VSE, RNN-VSE, T-VSE) with an embedding size of 256, using the following procedure and parameters.

CNN Model. We used an ImageNet pre-trained DenseNet 169 [4] with input image size 227×227 as the image encoder.

Image Data Augmentation. At training time, first resize to 333×333, then apply random crop of 227×227 and random horizontal flip with probability 0.5. At test time, first resize to 333×333, then center crop.

Two-stage Training. Stage 1: Freeze the convolutional layers of the pre-trained CNN, train the embedding layer and all the text encoder from scratch, for 2 epochs, with Adam optimizer, initial learning rate of 10^{-4}, reduced by half after 1 epoch. Stage 2: Train the whole VSE model for a maximum of 30 epochs, with Adam optimizer, initial learning rate of 4×10^{-5}, reduced by half after 5 and 10 epochs. Evaluate the model on the validation set after each epoch and save the best model.

Note that freezing the pre-trained CNN in stage 1 is crucial, otherwise the model does not converge. Lastly, we trained the text encoders from scratch, since the new vocabulary invalidates all the weights of the pre-trained models.

Loss Function. Symmetric triplet loss (max of hinges) with hard negative mining (max of hinges), with a margin value $m = 0.2$, as described in Section 2.1.

Batch Size. We used a batch size of 256 so that T-VSE model can fit on 4 NVIDIA V100 GPUs, each with 16GB memory, during training (at test time, a single V100 GPU is sufficient to compute both image and text embeddings concurrently with a batch size of 512).

3. Experiments

We trained all three models (T-VSE, AVG-VSE, RNN-VSE) on the training set of 11M product (image, title) pairs, evaluated on the 100K validation set, and tested the best model on the 1M test set.

For evaluation, we used the cosine distance to match each title/image to all the images/titles of the test set. As is common practice [3], we evaluated using R@K (Recall at K): the fraction of queries for which at least one correct result is returned in top K. We assume that for each query title/image, there is only one relevant image/title in the test

[2] We are planning to release the AF12M dataset.
[3] https://github.com/huggingface/transformers
[4] https://github.com/google/sentencepiece

Model / R@K		1	10	50	100
AVG-VSE	t2i	12.7	46.7	73.0	81.7
	i2t	8.4	31.3	52.4	61.2
RNN-VSE	t2i	12.7	47.3	74.5	83.3
	i2t	8.1	29.6	50.9	60.9
T-VSE (6 layers)	t2i	**30.7**	**76.6**	**91.9**	**95.1**
	i2t	**32.7**	**78.6**	**92.8**	**95.8**

Table 1. Text-to-image (t2i) and image-to-text (i2t) retrieval results on 1M test set of fashion product images-titles. Vocabulary size: 30K. Training epochs: 2+30.

Model / R@K		1	10	50	100
T-VSE (6 layers)	t2i	34.5	80.8	93.3	95.9
	i2t	36.6	82.2	94.1	96.4
T-VSE (12 layers)	t2i	**38.1**	**83.3**	**94.1**	**96.3**
	i2t	**40.5**	**85.0**	**94.8**	**96.8**

Table 2. T-VSE with 6 and 12 transformer layers, larger batch size (400) and longer training (2+50 epochs).

set, although there are some duplicate and near-duplicate products in the dataset that hurt the performance.

Table 1 presents the R@K accuracy on the 1M test set for the three VSE models, all trained for $2 + 30$ epochs (Sec. 2.5). T-VSE outperforms the other two models by a large margin on both text-to-image and image-to-text retrieval.

Table 2 presents two more experiments that further improve the performance of the T-VSE model. (i) We further trained T-VSE for 20 more epochs with a larger batch size of 400. We trained T-VSE with a 12-layer DistilBERT model for $2 + 50$ epochs with batch size 400. The results show that text model matters a lot in VSE, as well as the scale of the dataset. Transformer models can better leverage large training sets. They also benefit from longer training before the model saturates, as they have larger capacity.

AVG-VSE and RNN-VSE perform similarly, and image-to-text works worse than text-to-image, even though the loss function is symmetric. In T-VSE, image-to-text works slightly better than text-to-image at all recall levels; probably due to the more powerful text model.

4. Conclusions and Future Work

We showed that properly trained transformer-based visual semantic embedding (T-VSE) models achieve vastly superior results on cross-modal image/text retrieval, compared to classical VSE models that employ RNNs or simple word average. We experimented with the DistilBERT model, but other transformer models should also work as well or even better.

Finding enough parallel image-text data to train these data "hungry" models is challenging, but the e-commerce sites have plenty of such data readily available. Hence, T-VSE training can also be used as a pre-training step for many other problems. As a future work, it would also be interesting to explore unsupervised pre-training strategies for the transformer on product titles and descriptions, before the joint VSE training.

References

[1] Andrea Burns, Reuben Tan, Kate Saenko, Stan Sclaroff, and Bryan A Plummer. Language Features Matter: Effective Language Representations for Vision-Language Tasks. In *ICCV*, 2019. 1, 2

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2019. 1, 2

[3] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. VSE++: Improving Visual-Semantic Embeddings with Hard Negatives. In *BMVC*, 2018. 1, 2, 3

[4] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017. 3

[5] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *ACL*, 2018. 3

[6] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *EMNLP*, 2018. 3

[7] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *ICLR*, 2019. 2

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized bert pretraining approach. *arXiv:1907.11692*, 2019. 2, 3

[9] Priyanka Nigam, Yiwei Song, Vijai Mohan, Vihan Lakshman, Weitian Ding, Ankit Shingavi, Choon Hui Teo, Hao Gu, and Bing Yin. Semantic Product Search. In *KDD*, pages 2876–2885, 2019. 3

[10] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *NeurIPS Workshop*, 2019. 2, 3

[11] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *ACL*, 2015. 3

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. In *NIPS*, pages 5998–6008, 2017. 1, 2

[13] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, pages 5754–5764, 2019. 1, 2