Analysis of Bag-of-n-grams Representation's Properties Based on Textual Reconstruction

Qi Huang
New York University
qh384@nyu.edu

Zhanghao Chen
New York University
zc807@nyu.edu

Zijie Lu
New York University
zl1298@nyu.edu

Ye Yuan
New York University
yy1650@nyu.edu

Abstract

Despite its simplicity, bag-of-n-gram representation has not received much attention from research community in recent years. Empirically it has been proven to excel in many NLP tasks, therefore a further analysis on bag-of-n-grams representation's properties is deemed necessary. This general research question can further be divided into 1) To obtain a good distributed representation of n-gram and the subsequent bag-of-n-grams sentence representation, and 2) To analyze bag-of-n-grams sentence representation's properties with downstream tasks. This paper proposes a novel method to end-to-end train a recurrent neural network (with/without attention) on sentence reconstruction task to obtain n-gram embeddings, summing the resulting embedding vectors as bag-of-n-gram sentence representation, and analyze its properties with attention module's weight distribution and model performance on downstream tasks such as length prediction, word content predictions and machine translation.

1 Introduction

Though simple as it appears, bag-of-n-grams representation of textual data has been found to excel in many NLP tasks, in particular for sentiment analysis (Cho, 2017). This suggests that it may contain most information of a sentence's content. This paper focuses on investigating the properties of bag-of-n-grams representation, specifically whether it contains enough information to reconstruct the original sentence, and how the performance of sentence reconstruction correlates with that of the downstream tasks, thus inferring on its potential as a sentence representation.

To understand a sentence's meaning is a prerequisite for performing various natural language processing (NLP) tasks. This calls for a good representation of the meaning of a sentence. A wide variety of methods have been developed to generate a good sentence representation. Continuous-bag-of-words (CBOW) model (Mikolov et al., 2013) are efficient to train and performs well in many downstream tasks. However, it discards the word order information and some semantics. Recently, neural network based sentence representation models including Recursive Neural Network (RecNN) (Socher et al., 2012), Convolutional Neural Network (CNN) (Kim, 2014) and Recurrent Neural Network (RNN) (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014) have shown advantages in generating general purpose sentence representation. They capture more syntactic and semantic structures, but are all based on word-level embedding, and heavily rely on the model to figure out dependencies pattern, and are computationally heavy.

On the other hand, bag-of-n-gram representation models represent a sentence with a vector produced by summing or averaging over the pre-trained n-gram embeddings, incorporating more local order and syntactic information than CBOW yet still computationally cheaper than neural network based sentence representation models.

To build a bag-of-n-gram based sentence reconstruction model and produce n-gram based sentence representation, we use a simple average on n-gram embeddings generated from the sentence as its representation, and directly feed it to the decoder network for downstream tasks. We then end-to-end train vector representations of n-grams on sentence reconstruction task. A variation is to add attention module to the original model, let decoder attends to different sub-sum of n-grams embedding categorized by different value of n, thus creating a more dynamic training mechanism.

Despite the possible imperfect reconstruction result, the trained bag-of-n-grams representation is expected to encode a large amount of information after finishing the sentence reconstruction task. To further analyze what is contained in bag-

of-n-gram representation, especially the impact of choice of n, we further suggest to test our obtained sentence representation on various downstream tasks. We average n-gram embeddings in a sentence using learned matrix to obtain its representation for downstream tasks. By comparing the downstream task results and reconstruction results and analyzing the attention weight distribution on the sub-sum bag-of-n-gram vectors, we are able to better understand n-gram sentence representation and its potential.

2 Related Work

Recent works (Sutskever et al., 2014; Cho et al., 2014) have developed Encoder-Decoder models for machine translation. An encoder network reads and encodes the source sentence into a vector, and a decoder network then outputs a translation using the encoded vector(s) as input. When the source language and the target language are the same, the model becomes an autoencoder, and can be used to generate effective semantic representation of sentences or paragraphs (Li et al., 2015; Oshri and Khandwala, 2015). Nonetheless, these papers do not address the potential of using the encoder-decoder framework to generate optimal embedding vector. Following this line, our work makes the complementary contribution by suggesting using decoder and sentence reconstruction task to obtain n-gram distributed representation, and the bag-of-n-gram sentence representation from it.

On the other hand, word-level distributed representations and their corresponding properties have been analyzed extensively. In contrast, we observe that study on n-gram and its corresponding properties has been limited.

While bag-of-n-grams model is usually considered deficient in dealing with data sparsity and poor generalization in studies such as (Le and Mikolov, 2014), little work has been done to systematically analyze the correlation between the choice of n and the bag-of-n-grams embedding.

A few studies do suggest different approaches to embed bag-of-n-grams. For example, Li et al.(2017) propose a collection of methods to train a distributed n-gram representation in Neural Bag-of-Words. Essentially, all three methods mentioned in (Li et al., 2017) use an independent, simple linear layer to train and produce the n-gram embedding, mostly varying on their loss functions design (context-guided based, text-guided based, and label-guided based). Their work sheds light on generating n-gram distributed representation. Nevertheless, it does not adequately address the rationale underlying their model design, as the independent n-gram embedding network seems not to be a general-purpose module. Their analysis also stops at revealing differences in performance of different models, and does not make the connection between performance difference with n-gram per se.

Comparatively, our work provides a more fine-grained analysis of bag-of-n-grams representation, and produces the distributed n-gram representation using existing model, trained with an end-to-end fashion. We measure the amount of encoded information in the resulting bag-of-n-grams sentence embedding by sentence construction and other downstream tasks. The methodology we adopt is more systematic and can be applied to other sentence embedding model.

3 Approach

In this section, we aim to inspect the bag-of-n-grams embedded vector in a sequential task manner. The main idea of our method is to train the bag-of-n-grams embedding vector with the sentence reconstruction task, and feed it as raw input to downstream tasks. Then we measure the optimal performance that the classifier can achieve in each task, and investigate its relationship with the choice of n in both proposed models. The basic premise here is that the bag-of-n-grams embedding vector after sentence reconstruction task has been trained to preserve the most information. Despite the potential imperfect reconstruction result, the bag-of-n-grams representation is still expected to encode information such as sentence length, word presence and even semantics. This methodology is inspired by the work done by (Adi et al., 2016), we add a more challenging downstream task - machine translation - to test the overall performance of applying the trained sentence representation in a more practical manner.

3.1 Train bag-of-n-gram representation with Sentence Reconstruction Task

We introduce two models to train the bag-of-n-grams embedding vector.

Notation

Let S denote a sentence, and we use w_i^j to represent the j-th i-gram of the sentence. There are in total N_i i-grams of the sentence, and the i-gram representation of the sentence is $S_i = \{w_1^1, w_1^2, \cdots w_1^{N_1}, \cdots, w_i^1, w_i^2, \cdots w_i^{N_i}\}$. The i-gram w_i^j is associated with a K-dimensional embedding e_i^j, and so the vector of bag-of-i-gram representation of a sentence will be $E_i = \sum_{j=1}^{N_i} e_i^j$. After normalization, the bag-of-n-grams vector representation of a sentence $\bar{E}_i$ is given by

$$\bar{E}_n = \frac{\sum_{i=1}^{n} E_i}{\sum_{i=1}^{n} N_i}$$

Model 1: Decoder with Bag-of-n-gram as encoder

Inspired by RNN Encoder-Decoder model proposed by Cho et al (2014), we replace the encoder in our framework with a simple embedder that transforms a sentence S to its bag-of-n-grams vector $\bar{E}_n$. We maintain the general structure of the decoder, which is an RNN that is trained to generate a sequence of words by predicting the next word y_t given the hidden state $h_{\langle t \rangle}$ and its previous word y_{t-1}. The initial hidden state of the decoder $h_{\langle 0 \rangle}$ is the bag-of-n-grams vector output by the embedder, and the initial input y_0 is the starting of sentence (SOS) token.

Model 2: Decoder with Attention to different bag-of-n-gram vector representations

Instead of simply taking the normalized sum of all n-grams embedding and use the resulting vector as the sole initial input, we further propose to incorporate attention in our sentence reconstruction model. By splitting the original bag-of-n-grams vector by the size of n, we will then have n K-dimensional vectors. Then the model learn to "attend" to different sub-sum vectors during training.

Specifically, if we let $\tilde{E}_i$ denote the sub-sum vector of the i-grams representation of the sentence, then

$$\tilde{E}_i = \frac{E_i}{N_i}$$

Then the original bag-of-n-grams representation $\bar{E}_n$ will be split into

$$\mathcal{E}_n = \{\tilde{E}_1, \tilde{E}_2, \tilde{E}_3, \cdots, \tilde{E}_n\}$$

We also denote the set of index of sub-sum vectors in above as J.

When reconstructing sentence, decoder still outputs token of word y_i, and condition its output on previous outputs and an attention context x_i that consists of the current hidden state h_i and the attention module's output a_i. Following the implementation of attention mechanism in (Bahdanau et al., 2014), we let

$$p(y_i \mid \{y_1, y_2, \cdots, y_{i-1}\}, x) = g(y_{i-1}, h_i, a_i)$$
$$h_i = f(h_{i-1}, y_{i-1}, a_i)$$
$$a_i = \sum_{j=1}^{|\mathcal{E}_n|} w_{ij} \tilde{E}_j$$
$$w_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{|\mathcal{E}_n|} \exp(e_{ik})}$$
$$e_{ij} = t(h_{i-1}, \tilde{E}_j)$$

In above, g is a fully-connected layer with nonlinear activation function (here we use tanh) that takes last decoder output y_{i-1}, current hidden state h_i, and current activation output a_i as input. Current hidden state h_i is calculated with another fully connected layer f, which takes last hidden state h_{i-1}, last decoder output y_{i-1}, and current activation output a_i as input. The activation output is calculated by summing up the weighted sub-sum of bag-of-n-gram representations categorized n, weighted by attention energy e_{ij}. Finally, the attention energy e_{ij} is calculated by applying another linear layer t, with previous hidden state h_{i-1} and the corresponding sub-sum vector of n-grams. This model could be trained end-to-end nicely, and we expect to use attention mechanism to 1) mitigate the potentially high generalization error problem faced by the bag-of-n-gram-Decoder model and 2) probe the optimal size n_{optim} by picking the $\tilde{E}_i$ that corresponds to highest attention weight $\max_{w_{ij}}, j \in J$.

3.2 Downstream Task – Sentence Length

This task aims at evaluating how well the bag-of-n-grams representation can preserve the sentence length information. Given the bag-of-n-grams vector $\bar{E}_n \in \mathbb{R}^K$, the classifier's job is to predict the length of the sentence. We formulate this task as a multi-class classification, with several output classes according to preset length range.

3.3 Downstream Task – Word Content

This tasks measures how well the bag-of-n-grams representation can preserve the information of

each word token. Given the bag-of-n-grams vector $\bar{E}_n \in \mathbb{R}^K$, and a word representation $e \in \mathbb{R}^d$, the classifier's job is to determine whether the corresponding word w is contained in the sentence S. We formulate this task as a binary classification problem.

3.4 Machine Translation

This task measures the total amount of syntactic and semantic information encoded in bag-of-n-grams representation. The assumption here is if the bag-of-n-grams vector is that a sentence representation with sufficient information encoded is expected to have a relatively good performance on translating sentences. Given the bag-of-n-grams vector $\bar{E}_n \in \mathbb{R}^K$, the classifier's job is to generate a sequence of words $\{y^1, y^2, \cdots, y^N, EOS\}$ in another language.

4 Experiment Settings

4.1 Dataset

We perform experiments on the Tab-delimited Bilingual Sentence Pairs (English-French) dataset, which consists of 135842 English-French sentence pairs with various lengths.

Following Kyunhyun Cho's basic experiment setting, the training dataset is lowercased in order to avoid an issue of data sparsity (Cho, 2017). SpaCy is used for automatic tokenization. A shortlist of 30,000 most frequent n-grams are used to train our models.

4.2 Sentence Reconstruction

We train three types of models. The first one is an RNN autoencoder (RNNautoenc), which serves as a baseline. The second one is the proposed decoder with Bag-of-n-gram as encoder model, to which we refer as NgramDec. The last one is the proposed decoder with Attention to different bag-of-n-gram vector representations model, to which we refer to as NgramAttnDec.

All the encoder and decoder networks are Gated Recurrent Units (GRU) networks (Cho et al., 2014) with 256 hidden units each. In all cases, we use a multilayer network with a softmax activation layer to compute the conditional probability of each target word.

We use a stochastic gradient descent (SGD) algorithm to train each model. Random teacher forcing enabled with a probability of 0.5 is used to make the models converge faster. Each model is trained for 75000 iterations.

To measure the closeness of our reconstructed sentences and the original sentences from where n-grams are drawn, we adopt ROUGE (Lin, 2004) and clipped BLEU (Papineni et al., 2002) as our scores, with emphasis on recall and precision respectively.

4.3 Downstream Tasks

Multi-layer perceptron (MLP) models are used for classification in the length task and the word-content task.

For the machine translation task, we directly feed the the bag-of-n-gram representation vector into a decoder network to output a translation. The same training strategy described for sentence reconstruction is adopted. We then compare the ROUGE (Lin, 2004) and clipped BLEU scores (Papineni et al., 2002) of our translation with the target translations, and compare it with the RNN encoder-decoder model results (Cho et al., 2014).

Collaboration Statement

Zhanghao Chen works on Experiment Settings, Related Works and codebases. Qi Huang works on Research Question Framing, Model Design, Approach, Related Works and Prototype Experiment. Zijie Lu works on Approach, Introduction and Related Works. Ye Yuan works on Approach, Introduction and codebases.

The codebase of this project is publicly available at

https://github.com/HQ01/BOWMIAN

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv:1608.04207*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv:1409.0473*.

Kyunghyun Cho. 2017. Strawman: an ensemble of deep bag-of-ngrams for sentiment analysis. *arXiv:1707.08939*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv:1406.1078*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *Proceedings of the 2014 joint conference on empirical methods in natural language processing and computational natural language learning*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. *International Conference on Machine Learning*.

Bofang Li, Tao Liu, Zhe Zhao, Puwei Wang, and Xiaoyong Du. 2017. Neural bag-of-ngrams. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv:1506.01057*.

Chin-Yew Lin. 2004. Bleu: a method for automatic evaluation of machine translation. *Text Summarization Branches Out*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Barak Oshri and Nishith Khandwala. 2015. There and back again: Autoencoders for textual reconstruction. *Unpublished*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Rouge: A package for automatic evaluation of summaries. *Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics*.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. N. 2012. Semantic compositionality through recursive matrix-vector spaces. *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*.