Machine Learning System for Identifying Threats in e-commerce Chats/Calls

Richard Chang Tin Nguyen Bryn Tierney Lin Tao Steven Rodriguez
{richard_chang, tin_nguyen, bryn_tierney, lin_tao, steven_rodriguez}@intuit.com

Abstract

Today, it's a common practice for an e-commerce business to provide online support to its customers via chats and calls. Protecting customer data from potential threats is of paramount importance for business operations and customer success.

Traditional methods such as keyword searching or human inspection can be error-prone or expensive, and haven't scaled well, given the exponential increase in the quantity and length of the transcripts in recent years.
A common challenge for NLP-based threat detection applications is that the amount of labels can be sparse.

In this paper, we propose using state-of-the-art machine learning (ML)-based natural language processing (NLP) models to detect potential threats. We also introduce new ML/NLP methods to generate synthetic labels, and apply them to training the ML models.

1. Introduction

With the advance of technology, it has become common practice for many e-commerce businesses to provide online platforms to let customers and company agents interact with each other using online chats and calls. The global pandemic further boosted this online practice across all industries, which significantly increased the volume of chats and calls.

Addressing the potential risk to customer data, such as personal financial information, is of paramount importance for business operations and customer success. To provide a safe, secure environment for customer communications with a company's agents or partners, it is critical to proactively identify potential risks in chat/call transcripts. Most traditional methods for detecting risk rely on keyword-matching rule systems and human inspection. However, traditional keyword-matching techniques are not sufficient for detecting threats expressed by language, which can be a combination of words, sentiment, and contextual expression across multiple sentences in a particular chat or call. And, human inspection is costly and difficult to scale, given the exponential increase in the quantity and length of transcripts in recent years.

In this paper, we present ML systems that can detect the presence of threats, efficiently and cost-effectively. The systems fall within the broad field of intent

classification. The proposed ML system uses a combination of NLP models based on Transformer/BERT/XLNet models[1,2,3] and DNN(deep neural network.

Following are brief descriptions of the key concepts of those models:

1. Transformer[1]: is a deep learning model that adopts the mechanism of self-attention, differentially weighting the significance of each part of the input data.
2. BERT[2]: Bidirectional Encoder Representations from Transformers, is applying the bidirectional training of Transformer.
3. XLNet[3]: XLNET is a generalized autoregressive model where the next token is dependent on all previous tokens.

Those models provide different functionalities such as sentence embeddings, feed-forward networks, and autoregressive models to detect the contextual meaning of the transcripts. We also use Transformer/BERT-based models to create synthetic labels. Given some types of threats happened less frequently. The synthetic labels will help train the model efficiently.

2. Proposed Methods

We propose an ML system to detect potential risks in chat/call transcripts. The system can be generalized to other business use cases with different training data.

[1] https://arxiv.org/abs/1706.03762
[2] https://arxiv.org/abs/1810.04805
[3] https://arxiv.org/abs/1906.08237

2.1 Feature engineering

Feature engineering is one of the most important steps in building a successful ML system. It requires applying advanced ML methods to examine input data, and extract information(feature) based on the data. In practice to build an effective ML system, we applied the following 2 ways described in detail below to better improve the data quality before it is fed to train the ML system. Those two methods help narrow down the right data set for training as well as improve the balances between negative and positive samples of the data.

A. Regular Expressions:

Regex is one of the most important concepts in NLP. [11]. Since it is rule-based and human-generated, regex is very intelligible, concise, and easy to tune. This is why it is very widely used in the industry for various tasks such as pattern matching, entity extraction, and in our case, information filtering.

However, the flip side of regex reliance is it is very hard to design an abstract regex pattern general enough given the complexity of human written and spoken language.

Subsequently, in our application, we combine it with the use of deep neural network (DNN) algorithms where a broad set of carefully written and tested regexes is used to limit the amount of context and pattern. This method will help the DNN better learn each dialogue's meaning.

Another upside of having this filtering layer is to improve the data quality for sparse datasets. As we know from the dataset, positive labels are very hard to come by.

Thus filtering our non-application relevant conversation such as greetings, introductions, etc., improves the ratio of positive labels in the final dataset. This along with sampling and label generation techniques discussed later in the paper helps us achieve an 8-10X improvement in case detection rates.

Base Sentence	Observed Sentence	Cosine Distance Score
I'm gonna send you an email	I'm going to send you an email	0.994858
	I'm gonna send an email to you	0.991155

Table 1: Cosine distance (similarity) score to address human language variation. The score ranges between 0 and 1. Identical sentences would have a score of 1.

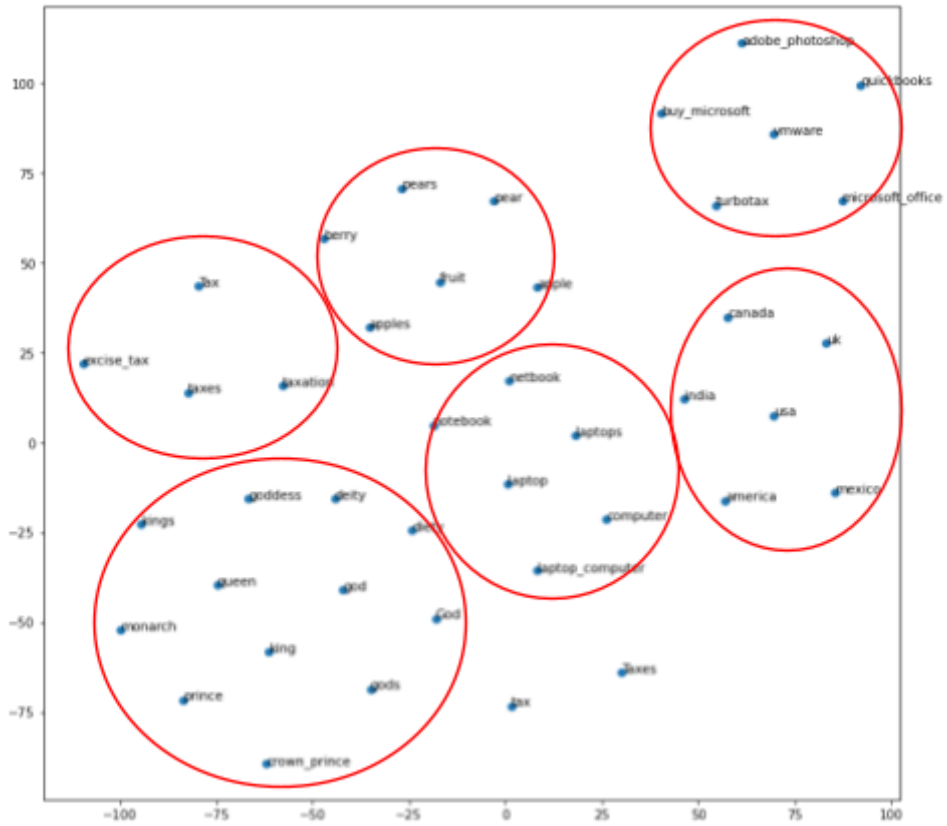

Figure 2.1.1: Example of word similarity

B. Text similarity:

Another filtering and feature engineering technique we applied was using similarity on word embeddings.

Embeddings in NLP are the dense numeric vector representation of words in low dimensional space. The very first proposal was published in 2013 by a team of researchers at Google [13]. Since the usage of word embedding and similarity has been actively used in many NLP applications. The reason for this is its strength. Unlike traditional dictionary-based embedding and word frequencies. Deep learning word embedding is able to map the context and semantic information of a word in a numeric form and allows practitioners to perform mathematical operations on. An example of this is:

vec(“Madrid”) - vec(“Spain”) + vec(“France”) is very close to vec(“Paris”) [14]

Another example is shown in figure 2.1.1 using word-embedding vectors, we can clearly make out cohorts of words that are similar to each other. Using this technique helps us limit the number of samples needed for the later classification task thus limiting the pattern our model has to learn.

This was first applied on the word2vec model but more advanced models like BERT would also take contextualized meaning into account which makes it even more powerful for NLP tasks.

In order to achieve good results, we:

a. Generate a base sentence pool. This part includes (1) grouping useful sentences from known business cases and positive cases (Query) and (2) pulling transcripts from a short time range in the past. In this case, we pulled transcripts from the past 12 months (Corpus). All sentences are embedded using pre-trained sentence BERT embeddings [10]. Embedding a short time range of transcripts helps with cost savings the system is applied to a large volume of transcripts. Next, the cosine distance is calculated between Query and the historical short-term transcripts. The sentences with the smallest cosine distance, together with the original Query sentences, make up the base sentence pool.

b. After generating the base sentence pool, a human review of the base sentence relevance is executed. This step is highly recommended to remove sentences that could cause a large volume of false positives. In production, any real-time transcripts can be embedded and a cosine similarity score is calculated between the real-time transcripts and the base sentence

2.2 NLP Labels Synthesization

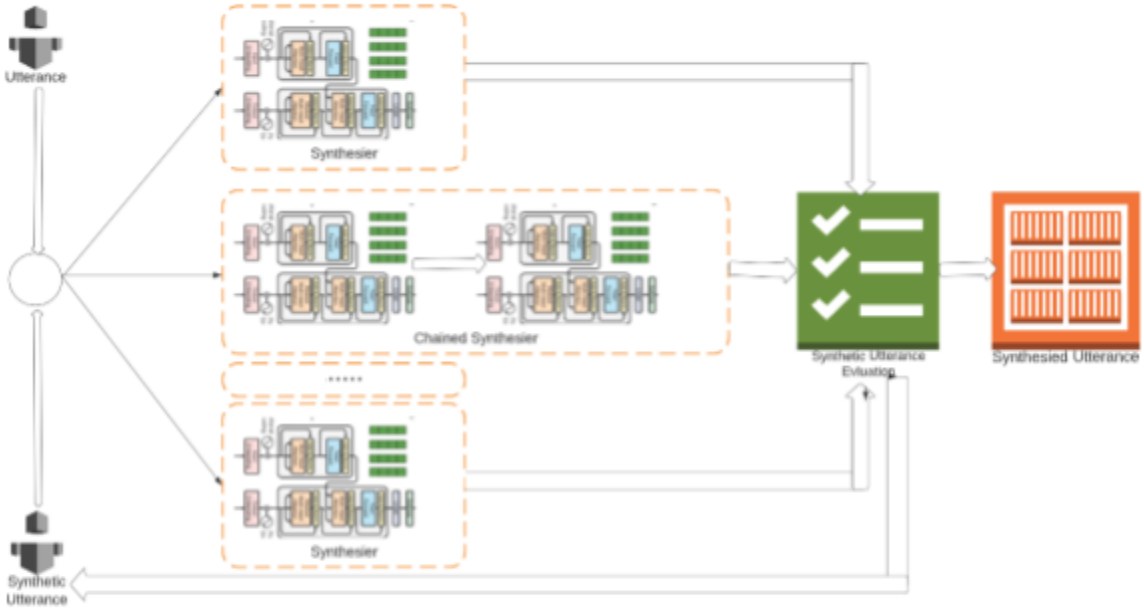

Fig 2.2.1 Framework to create synthesized NLP labels

To train an NLP model, it often requires a large set of labels to start with. This would be a challenge to get a set of labels of threats. This is mainly due to the fact as follows:

a) Some threats can happen less often and there is no data set existing.
b) Understanding the meaning of the transcript often requires understanding the context of the transcript.
c) Language itself is difficult given it often involves expression, sentiment, idioms, etc.

Because of the above reasons, it's often difficult to get good quality labels for certain threats. To solve this problem, we develop a framework to synthesize the NLP labels. The framework works as followings:

a. The framework is composed of a set of synthesizers
b. Each synthesizer can be a single model or a chain of Transformer based models to generate synthetic utterances based on the input utterance. The generated utterances have the same or similar contextual meaning as the input utterance does
c. After each synthetic utterance is created, an evaluation process, which is supported by another Transformer model, will evaluate the synthetic utterance. The evaluation process will eliminate those utterances that don't have a similar contextual meaning as the input utterance. The utterance passing the evaluation will become a label of the training. It can also be fed back to

the input of the framework to generate new synthetic utterances

The diagram of the synthetic utterance generation framework is shown in Fig 2.2.1.

By running this framework we can generate a set of labels in the quantity 30-100 times the existing labels. For some models, we started with 30-50 labels and ended up with more than several thousand labels. This would greatly improve the quality of the trained model.

2.3 Transformer-based models to detect threats

Since the invention of Transformer based models in 2017, those models have demonstrated significant improvement over the NLP models. In the previous section, we have already applied some of the Transformer models to create synthetic labels, and sentence embedding. We further build models based on the latest Transformer models to understand the contextual meaning of the chats/calls. In this paper, we will build our model based on XLNet.

After applying the methods stated in 2.1 and 2.2, the filtered and embedded transcripts will be sent to the ML/NLP model to detect the risk of fraud.

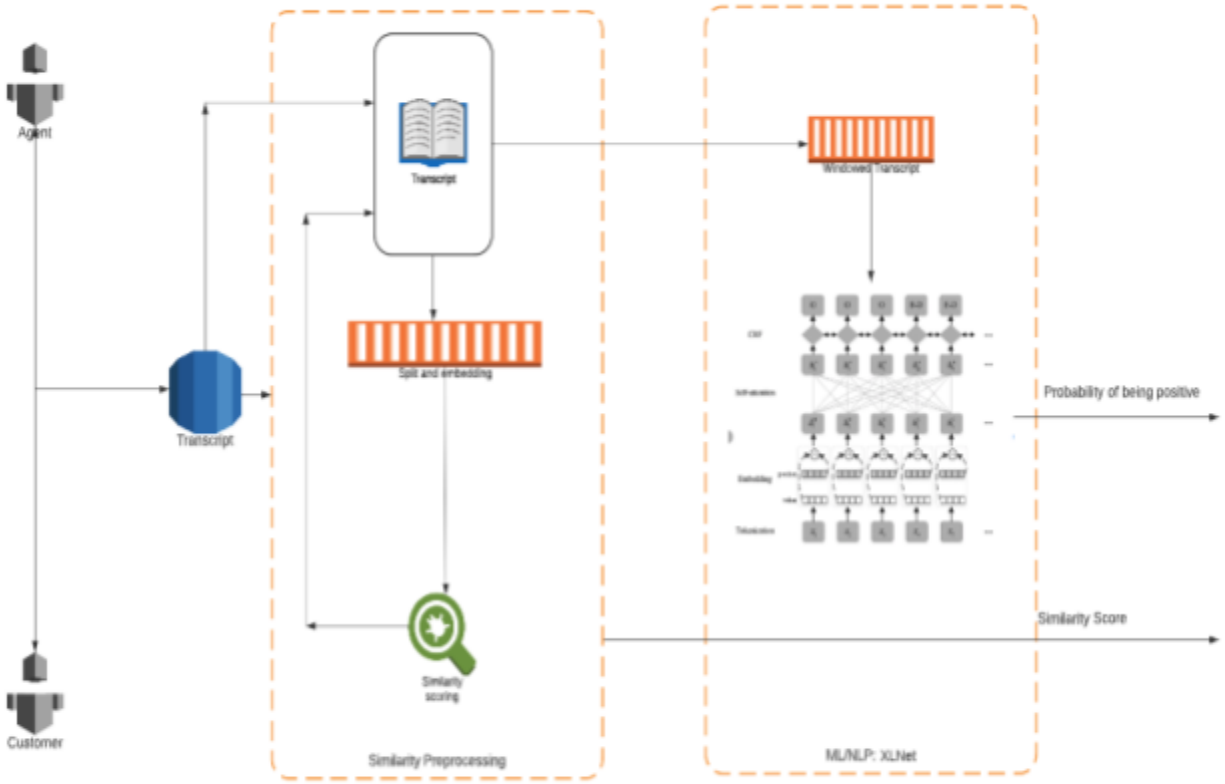

Fig 2.3.1 ML/NLP model architecture

Fig 2.3.1 is the diagram of a system built with the XLNet model.

The same architecture is used for both training and inference.

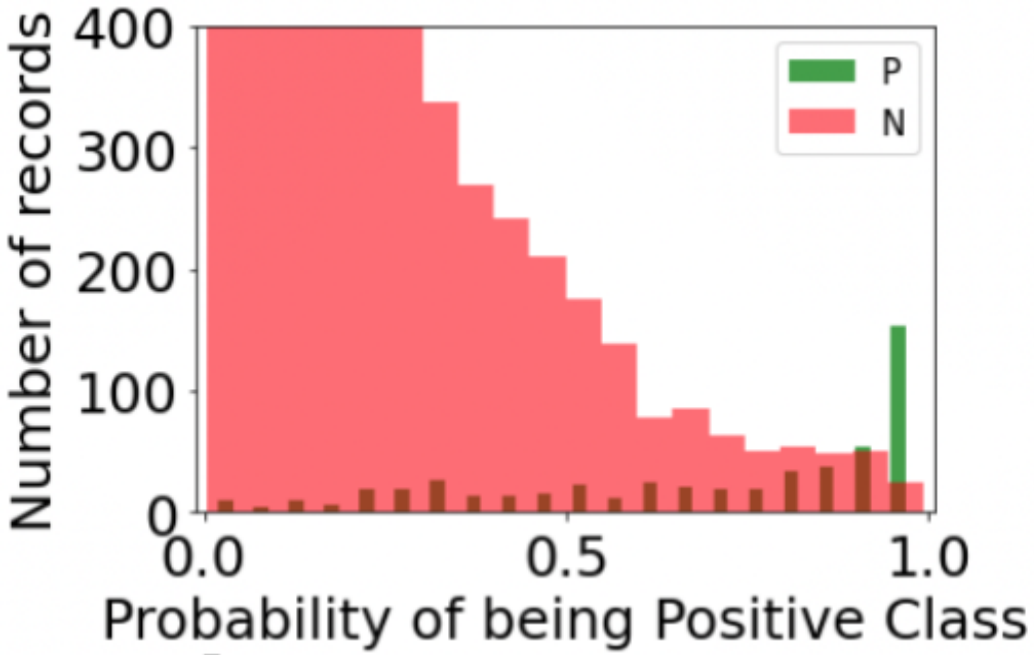

Fig 2.3.2 ML/NLP model trained to detect threats

In Fig 2.3.2 The chart shows the model score distribution of the samples of both positive and negative cases. The higher the score, the higher possibility that the transcripts contain threats.

Fig 2.3.3 shows the corresponding precision/recall chart of the model performance. The model has demonstrated a fairly good recall and precision rate overall. For example, with a threshold at 0.75, the precision and recall rate will be at 0.6.

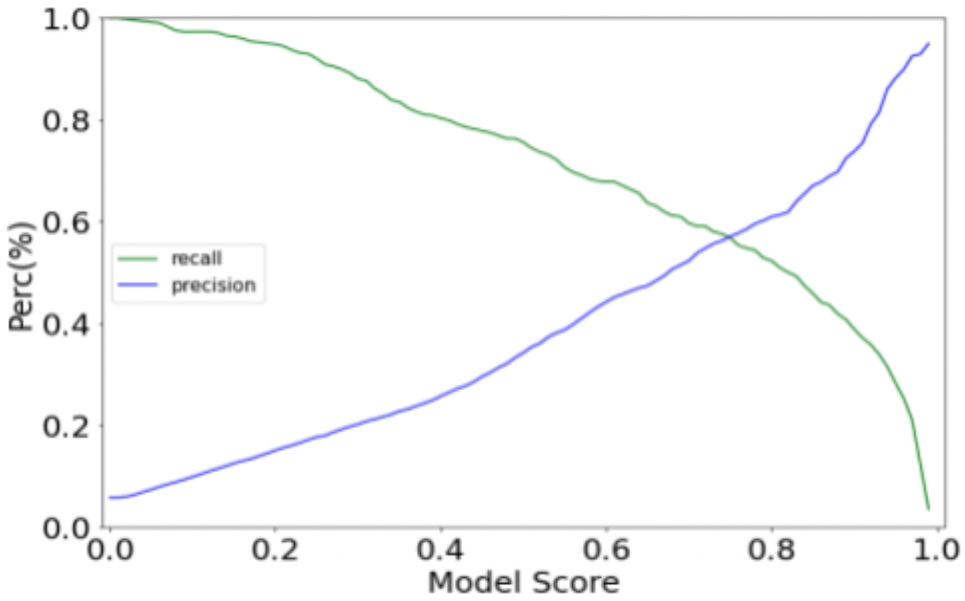

Fig 2.3.3 Model Performance Metrics

2.4 Human Feedback

Since this is an incredibly hard task with very limited positive labels of threats to start with, we also build a pipeline to allow humans to inspect the results from the ML models. The human inspection result will be used in further ML model training. By doing that, we can keep improving the ML system. Meantime, given the ML system has taken the majority load of examining the chats/calls, this method is scaling well without skyrocketing cost.

For our human inspection system, we have three sequentially looping steps:

1. The trained ML system examines chats/calls data and provides output as described above.
2. Every threat detection is sent to a human operator whose job is to investigate and give the best label for said detected threat.
3. The newly labeled data is incorporated with existing data to train the newer ML system which will end up with better performance compared with the previous version. classification, comparing this second system with the first system.

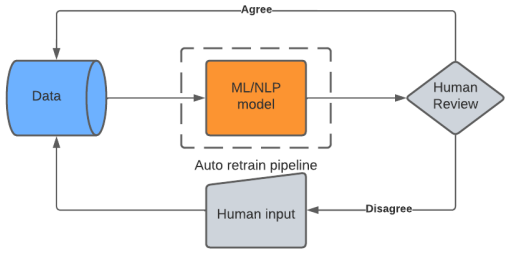

Fig 2.4.1 Human feedback system

3. Conclusion

We proposed a systematic approach to detect critical threats from chat/call transcripts based on cutting-edge ML and NLP technology. This approach is imperative for protecting customers' data security and companies' reputations. It replaced the previous system which is solely relying on keyword-matching rules and human inspection, which doesn't scale well for a high volume of chats/transcripts. The new approach is also much more precise than keyword matching in capturing errors expressed with complex language expressions.

The above solution, which is based on the latest ML and NLP models, is scalable and powerful to meet the high volume of demand of today's e-commerce business. The solution has been implemented and used in the daily production environment at Intuit. It has significantly contributed to proactively identifying and preventing threats in e-commerce chats/calls.

References:

[1] 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.

[2] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[3] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le, XLNet: generalized Autoregressive Pre Training

for Language Understanding, https://arxiv.org/pdf/1906.08237.pdf, 2020

[4]Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In Advances in neural information processing systems, pages 3079–3087, 2015.

[5] Yaoyu Zhang, Yuqing Li, Zhongwang Zhang, Tao Luo, Zhi-Qin John Xu, Embedding Principle: a hierarchical structure of loss landscape of deep neural networks, https://arxiv.org/abs/2111.15527, 2021

[6] William Fedus, Ian Goodfellow, and Andrew M Dai. Maskgan: better text generation via filling in the_. arXiv preprint arXiv:1801.07736, 2018

[7]Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. CoRR, abs/1409.0473, 2014.

[8] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. CoRR, abs/1406.1078, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 770–778, 2016.

[10] Shuailiang Zhang, Hai Zhao, Yuwei Wu, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. Dual co-matching network for multi-choice reading comprehension. arXiv preprint arXiv:1901.09381, 2019.

[11] Reimers, Nils, and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks." arXiv preprint arXiv:1908.10084 (2019).

[12] Chang, Angel X., and Christopher D. Manning. "TokensRegex: Defining cascaded regular expressions over tokens." Stanford University Computer Science Technical Reports. CSTR 2 (2014): 2014.

[13] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." Advances in neural information processing systems 26 (2013).

[14] Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv:1301.3781 (2013).