Extracting Semantic Features to Understand the Meaning of Words

Binit Gajera, Divyesh Chitroda
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
{bgajera1, d151} @umbc.edu

Abstract

Semantic features or concepts play an important role in understanding the meaning of a word in a given context. The features depend upon the dataset that provides context and has a sparse representation of the word as a concept. These semantic features can be either handpicked or generated by analyzing its co-occurrence with the other words. In this paper, we predict the semantic features of a concept(noun) type. We use the features of the McRae dataset for a given concept and create a mapping of those features to the embeddings obtained from the BERT model for that specific concept. Our goal is to decode the meaning of the BERT embeddings using the feature vectors of the McRae dataset.

1 Introduction

When we look or think of an object, in our mind, we know what the object looks like, what color it is, what size it is. We form a mental model of the object's perception, and when we explain the object by expressing our thoughts into language, we indicate various semantic concepts that the object represents. We can see why these features are important to understand the word by examining the relationship of a word with those features.

Using Natural Language Processing, we can extract the relationships of a word which results in semantic features that are mapped to neural activations. We look at both statistical(Church and Hanks, 1990) and neural(Cheng and Kartsaklis, 2015) learning approach to extract semantic features of a word. We aim to generate a dense semantic feature representation of a word that represents its meaning, or at the least, its context. The dataset mentioned in the following discussion allows us to create this dense representation and mapping of semantic features with particular concept types.

2 Motivation

This project is a part of a bigger project in which we would be analyzing fMRI images to decode neural activation in the brain for a word. To understand how the human brain decodes words and interprets the meaning of those words, Mitchell et al. has presented a novel approach. In the task of identifying and predicting brain activation for a particular stimulus(object), it is challenging to map a single word to the neural activations. They have mapped regions of brain activation to semantic features of the words(nouns) that were presented to the subjects.

Neural activation for a word is a linear function of semantic features of a word (Mitchell et al., 2008). It is crucial to identify how these semantic features are mapped to a target word. We do not know which brain region is decoding semantic features of the language, but we can identify and estimate these features by analyzing the word itself and the context in which it occurs. By examining (Sudre et al., 2012) and (Kivisaari et al., 2019), we can see that no matter the number of features, we can always model these features such that they map to some neural activation. Therefore, the more semantic features we can identify, the more we will be able to understand the meaning of that word and its related neural activation.

3 Related Work

One of the implementations that are performed by (Zahn et al., 2007) is very similar to what we tend to achieve. The authors represent only Social concepts based on semantic features. The paper (Mitchell et al., 2008) has also used semantic feature representation for neural activation prediction, which is well-recognized, as we discussed in the previous section.

For the specific task at hand, of generating se-

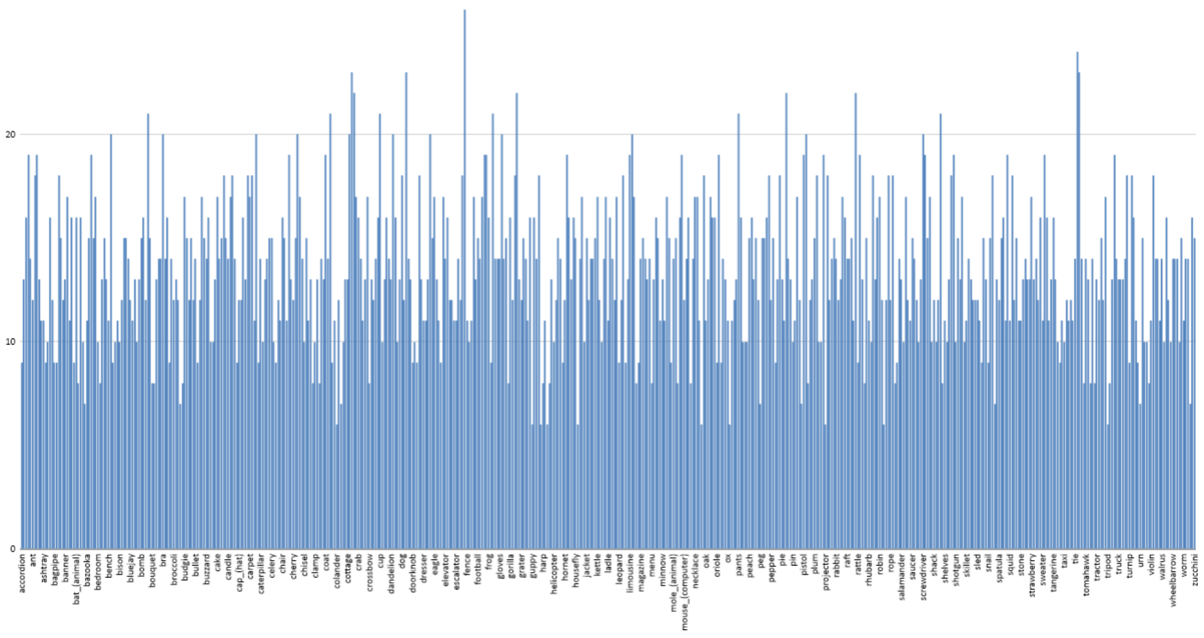

Figure 1: Distribution of Features(hypernyms) for Concepts(nouns) in McRae Dataset

mantic features, we must have a large vocabulary to generate a mapping between the tokens and the features. Initially, for such a requirement, we had a plan to consider the vastly used database in NLP that is (Miller, 1995), because WordNet consists of extensive vocabulary but has less feature mapping. Due to this limitation, we will use the dataset provided by (McRae et al., 2005) in which there is a variety of information available for a token, and thus that information can be used to generate a pattern or a sequence of mapping between different categorical tokens.

Other methods include identifying hypernym/hypernym relation using automatic feature discovery (Snow et al., 2005). The authors used a dependency parser to represent these relationships by identifying lexico-syntactic patterns between noun pairs. Their results show that a hypernym classifier trained using logistic regression performs better than Naive Bayes and other pattern-based approaches.

4 Proposed Solution

In contrast to Word2Vec, BERT (Devlin et al., 2018) embeddings do not represent an implicit set of a similar or closest cluster of words. This allows us to observe these embeddings in a different way to establish semantic relationships.

We build and train a neural network that predicts the features from the McRae dataset for the embeddings of a specific token generated from the BERT model. Here, the embeddings from BERT are floating values which we map to the corresponding possible features. Since the features and the embeddings are generated for the same token, there exists some correlation between the values, and to decode the encrypted embeddings from BERT, we use a 2-layer neural network approach.

We call the embeddings as encrypted because the literature has no significant release of information about those values from the authors of BERT. Since their model is a neural network that performs astonishingly well but the features behind each prediction are quite essential and undiscovered.

5 Approach

The McRae dataset consists of 541 nouns as 'Concepts,' which is collected from 725 participants and manually tagged with features (McRae et al., 2005). Each of these nouns is represented using a set of hypernyms that describes the semantic features of the noun. For example a concept 'airplane' is represented as *has_wings, used_for_passengers, is_fast, requires_pilots, used_for_transportation, found_in_airports, is_large, made_of_metal* and so on. Moreover, these concepts have other useful features such as WB_Label taxonomy label, according to (Wu and Barsalou, 2004), BR_Label, according to (Cree and McRae, 2003) "brain region taxonomy", several visual, sound, taste, functional features. We have restricted the scope of this project to hypernym features.

Our goal is to learn the mapping between BERT embeddings and McRae hypernym features to predict hypernyms for a noun. The BERT embeddings represent a token in a particular context with a 12 layered vector of 768 float values. These embeddings are subject to change for any given token and depend upon the context of the token. The embeddings represent semantic characteristics based on the meaning and context of a word. These embeddings can be directly used in a wide variety of NLP tasks. However, to decode the meaning of a word, these embeddings should say something about a word instead of just having numeric values.

5.1 Dataset

We are using the McRae dataset to get the feature names of the 541 types, figure 1 shows the distribution of 541 unique nouns in the dataset. The BERT model developed by a team at Google, which gives us the word embeddings in terms of floating-point values for the same types that we fetch from the McRae dataset. For our neural network, we have created a training dataset using which we can map the word embeddings produced from the BERT model to the feature names of the specific type of the McRae dataset. Our end goal here is to create a mapping between the floating-point values from the BERT model and the feature names from the McRae dataset. To achieve the task, we created a one-hot encoded vector for the unique features using the 'DictVectorizer' from scikit-learn. This vector is the target label that the model has to predict. We have split the dataset into 80-10-10 train, validation, and test set.

5.2 Model and Task Framework

For each of the nouns in the McRae dataset, we have generated BERT embeddings resulting in 541x12x768 feature tensor. This embedding representation is our training data. Then, we have vectorized each of the features for the concept nouns to represent the hypernym features as one-hot encoding resulting in a 541x2526 dense matrix. This binary dense hypernym representation is our target labels. We use a 2-layer neural network[1] to perform a multi-label classification of our training embeddings to predict classes that represent hypernym features as labels. We are using BCELoss and SGD optimizer to perform gradient optimization. Figure 2 shows the architecture diagram of our network.

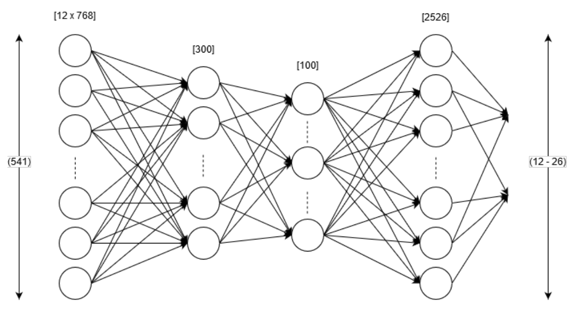

Figure 2: 2-layer NN Architecture for the task

6 Prediction and Evaluation

The prediction of our model is the probability of whether a feature fired or not using a threshold that is the mean probability of the predicted probability distribution. Since we have introduced the weights to our BCE loss function, the probability values are penalized because of which 0.5 no longer represents the center value, and hence we use the mean as the threshold. We create a one-hot vector of the prediction from the above approach and then evaluate it. Figure 3 shows the features that crosses the threshold(red line) for 6 concepts(nouns). The features(out of 2526) that lie to the right end of the threshold are the predicted label.

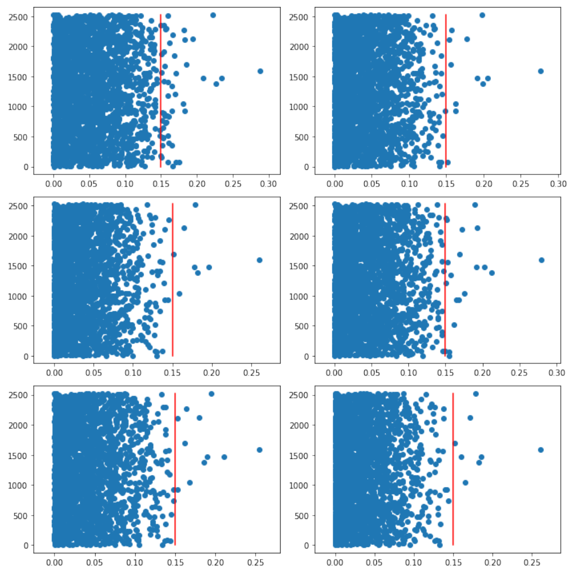

Figure 3: Multi-label prediction strategy

For the evaluation, we set a tolerance of 99%,

[1] https://github.com/div5yesh/semantic-feature-extraction

which signifies that if the prediction matches the target higher than 99%, then we label it as a correct prediction. But if the prediction of the labels matches less than 99%, then we label it as incorrect. We had to take this approach to predict the labels because our current neural network architecture is not able to predict all the features correctly. Given a more optimized loss function and probably a stronger neural network, we believe that the task can be achieved for all features, and the accuracy can also be improved significantly.

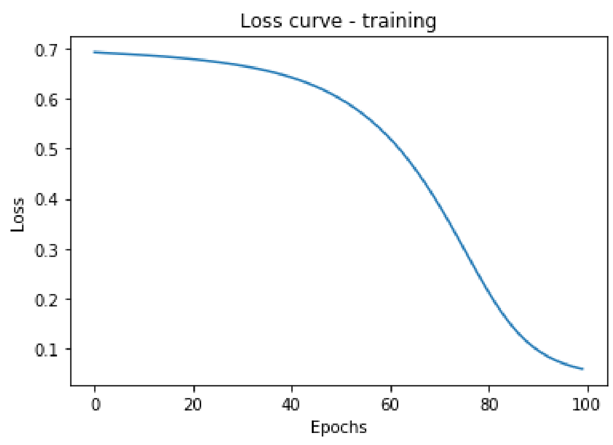

Figure 4: Training Loss Curve (Epochs vs Loss)

7 Results

Figure 4 show the training loss for 100 epochs of training in a batch size of 4 data points. The model predicts the most common features among all the concepts since our model is using Binary Cross-Entropy Loss, which maximizes the log-likelihood of 10 high-frequency features. The feature *made_of_metal*, *is_small* and *an_animal* has frequency of 133, 121 and 99 which are among top 50 features show in Figure 5. So the model predicts these 3 features for most of the concept embeddings, which can be seen in Figure 3 as the rightmost data point in all the plots.

Our model has an accuracy of 89.73%, recall of 10.89%, and precision of 17.09%. The metrics here suggest that the model was able to predict at least some of the McRae features for a specific type using the BERT embeddings. We experimented with weighted Binary Cross-Entropy loss that improved results, which means now the model focuses more on getting the maximum probability for the desired feature instead of just predicting all ones or zeros. Finally, we used weight equal to 3 for each positive label in the feature vector, which gave us the best and consistent results.

All the testing was done on the Validation data created using the 80-10-10 split of the data, and based on its results, we were able to fine-tune the model. The final scoring metrics shown are for the actual unseen test data, which is evaluated after complete training and fine-tuning of the neural network and loss function hyperparameters.

8 Challenges

The first and foremost difficulty that we faced was regarding the training of the neural network. Our model consists of Linear layers in which we have about 300 and 100 neurons on the layers, which makes the computation a bit expensive and time consuming to run on CPU. So, we modified our code to run it on the GPU.

The major challenge of designing a neural network for a completely new dataset is the "loss function". Here, we initially had used the LogSoftmax output function for the floating values obtained from the BERT model. Later, we converted the predicted feature values to the one-hot encoded vectors using Softmax. Since our classification task was multi-label multi-class classification, we changed the loss to Binary Cross-Entropy loss, which works explicitly for such vectors, and the model was successfully able to converge.

One other challenge that we face was with producing the word embeddings for the tokens that are present in the McRae dataset. The BERT model produces token, segment, and position embeddings, but we use only token embedding for the McRae concepts which are present in the BERT vocabulary. Upon further analysis, we noticed that some of the tokens were not present in BERT's vocab, since BERT tokenizer breakdowns words into WordPiece tokens. Eg.

$$ashtray = [\text{“}ash\text{”}, \text{“}\#\#tray\text{”}]$$

Because of such pre-processing, our neural network would learn different features for each of the partial token generated by BERT, so to avoid such conflict, we limit the model just to use the first partial token generated by BERT. We have a multi-label and multi-class prediction task for which we need an evaluation strategy that is capable of such analysis. The metrics such as Accuracy, Precision, and Recall available in 'sklearn' evaluates class-wise predictions instead of considering the

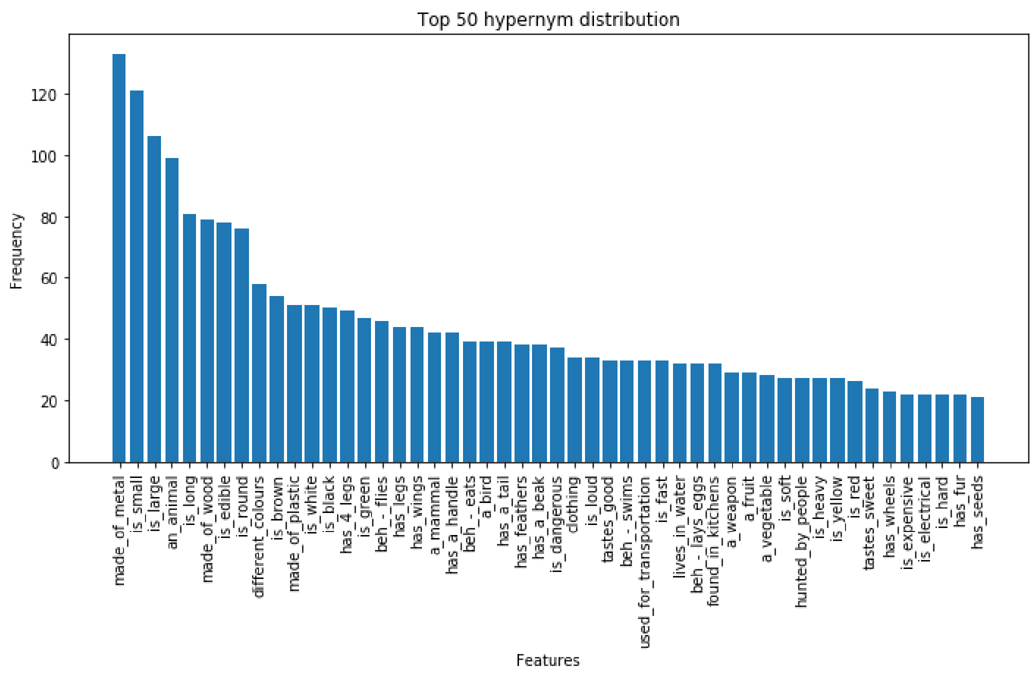

Figure 5: Top 50 hypernym features for McRae Concepts

complete one-hot vector as a multi-label prediction output.

Therefore, we implemented an evaluation function that matches the number of features for predicted and actual labels for a data point. We are matching the maximum number of features that the model predicts, and if the model predicts even 25 features correctly, we consider that as a true positive. Similarly, we calculate false positives and true negatives.

9 Future Work

Moreover, the task framework can be extended to use features beyond hypernyms from the McRae dataset. Since McRae dataset is rich in features, we can use other columns such as the number of letters, phonemes, syllables, concept familiarity, and intercorrelation density to train the model. For a probabilistic model such as MaxEnt, it would be beneficial to have more features. The BERT tokenization decomposes some of the words into smaller pieces following hashtags to create a WordPiece model to distinguish subwords from standalone types in the vocabulary. Due to this, we end up with multiple BERT tokens from a single McRae concept. To tackle this issue, we can use the mean of WordPiece embeddings to collapse them to represent a single McRae concept again.

Further, our model is sensitive to the context-dependent BERT embeddings. Since BERT generates different embeddings for each token in a particular context, relating them to McRae features does not make any sense. To overcome this, we would use a corpus of text that contains McRae concepts and use each of the unique contextual embeddings of a concept as a separate data point. This would increase the amount of training data for each concept without manual data acquisition. From the results, we see that our model's Cross-Entropy Loss is maximizing the probability of most common features, we would like to experiment with other loss functions like Hinge Loss or Class Rectification Loss (Dong et al., 2019).

References

Jianpeng Cheng and Dimitri Kartsaklis. 2015. Syntax-aware multi-sense word embeddings for deep compositional models of meaning. *arXiv preprint arXiv:1508.02354*.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

George S Cree and Ken McRae. 2003. Analyzing the factors underlying the structure and computation of the meaning of chipmunk, cherry, chisel, cheese, and cello (and many other such concrete nouns). *Journal of experimental psychology: general*, 132(2):163.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Q. Dong, S. Gong, and X. Zhu. 2019. Imbalanced deep learning by minority class incremental rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(6):1367–1381.

Sasa L Kivisaari, Marijn van Vliet, Annika Hultén, Tiina Lindh-Knuutila, Ali Faisal, and Riitta Salmelin. 2019. Reconstructing meaning from bits of information. *Nature communications*, 10(1):927.

Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris Mcnorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37(4):547–559.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Tom M Mitchell, Svetlana V Shinkareva, Andrew Carlson, Kai-Min Chang, Vicente L Malave, Robert A Mason, and Marcel Adam Just. 2008. Predicting human brain activity associated with the meanings of nouns. *science*, 320(5880):1191–1195.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in neural information processing systems*, pages 1297–1304.

Gustavo Sudre, Dean Pomerleau, Mark Palatucci, Leila Wehbe, Alona Fyshe, Riitta Salmelin, and Tom Mitchell. 2012. Tracking neural coding of perceptual and semantic features of concrete nouns. *NeuroImage*, 62(1):451–463.

L Wu and LW Barsalou. 2004. Perceptual simulation in property generation. *Manuscript submitted for publication*.

Roland Zahn, Jorge Moll, Frank Krueger, Edward D. Huey, Griselda Garrido, and Jordan Grafman. 2007. Social concepts are represented in the superior anterior temporal cortex. *Proceedings of the National Academy of Sciences*, 104(15):6430–6435.