METAPHOR DETECTION USING TRANSFER LEARNING

Krishna Surendra Palepu, Naveen Kumar Nettem, Sasidhar Reddy Katikam

UNIVERSITY OF NEW HAVEN

(kpale1, nnett2, skati5)@unh.newhaven.edu

Abstract

Metaphors are an important component of natural language as they make the language more creative. It allows people to apply their knowledge of the base, which is typically more concrete and familiar, to inform their understanding of the less-familiar target. While humans can easily understand multiple meanings behind a sentence, it is challenging for machines to comprehend the non-literal meaning behind the sentence.

Metaphors in nominal sentences are made up of two parts: a target, which is the topic of the statement, and a base, which provides information about the target. The use of poor choice of words and acronyms in online texts, posts and comments make it even harder to process natural language. This has triggered the industrial and research community in the last few years to come up with a solution for detection of metaphors. However, these steps are still in their infancy and need further development.

INTRODUCTION

Metaphor has been an essential difficulty of language and thought, it is an often used figure of speech. In everyday life, one can face metaphor in every three phrases. Apparently unconnected aspects of one notion are connected with another topic in metaphorical language. The advent of digital media has made the need of a metaphor detection system a necessity since it is challenging for machines to comprehend the intended meaning behind the sentence. Combining this with poor use of language and acronyms in online conversations makes it harder to process the natural language. Metaphors are nowadays used in many areas ranging from regular day to day conversations to important news articles. Hence an efficient model for Metaphor detection is needed to be build. This project deals with building a model that is designed for metaphor detection.

Metaphor detection has been a challenging aspect of natural language processing. Because of the abundance of electronic writings that have developed in the Digital times, metaphor identification has become a focus of NLP. The impact of metaphor recognition on NLP tasks such as reading comprehension, automated summaries and machine translation is immediate.

The existing models will be analyzed to figure out what must be done and where to put in more work to increase the efficiency of the model. Different models related to Natural Language Processing and metaphor detection will be tested out to see which one suits the best. The model will then be trained using a metaphor corpus. The model's performance will then be compared to the existing models and the necessary tweaks will be made. The model will be finally deployed for further testing.

Data Set

We will be using two datasets, the first dataset will be Birke and Sarkar (2006) TroFi Dataset. This dataset comprises of 50 distinct verbs in sentences. There are multiple literal and metaphoric sentences for each verb. There are a total of 3737 samples in this dataset.

For the second dataset, the VUA dataset by Pérez-Sobrino (2014). This dataset has sentences from different verbs. There are a total of 17380 samples made with 2063 unique verbs. The above two datasets are similar in nature with same type of columns

	verb	sentence	verb_idx	label
0	absorb	An Energy Department spokesman says the sulfur...	22	0
1	absorb	The yellow beta carotene pigment absorbs blue ...	5	0
2	absorb	This time , the ground absorbed the shock wave...	5	0
3	absorb	" Vitamins could be passed right out of the b...	12	0
4	absorb	As Eliot wrote : " In a warm haze , the sultr...	14	0

	verb	sentence	verb_idx	label
0	fail	Ca n't fail to be entertaining .	2	0
1	go	How much was he going to tell her ?	4	0
2	win	Up until that news hit the Committee , Don had...	10	0
3	go	Could go on to the rugby and go with them coul...	1	0
4	go	Finally , we went to the office and they gave ...	3	0

System Requirement Analysis

Through our model, we aim to separate the metaphorical and non metaphorical sentences from a given corpora. Our aim is to build a Deep Learning model that works towards detecting metaphorical sentences by analyzing the text of the sentence using NLP.

Functional Requirements:

Our model should be able to verify the metaphorical nature of sentences and provide the classification of sentence in terms of 'literal' or 'non-literal'.

Any python 3 installed device should be able to run the model

Non-Functional Requirements:

To run the algorithm a user must need at least 8GB of ram or any cloud computing or online service such as google Collaboratory.

A good GPU is recommended to make the model perform calculations faster or to train the model faster.

Visualization

Count plot : For checking the balance of datasets, number of sentences for a list of verbs, number of literal and non-literal sentences based on sentence polarity.

Histogram : A histogram is basically used to represent data provided in the form of specific groups. It is used for text length vs number of texts. It is most trusted method for visualization of numerical data distribution.

Word cloud : Visualization process used to represent text data. where each word's significance ad frequency are indicated by its size larger the word in plot more important it is. Those which are highlighted are more significant.

Horizontal Bar plot : Horizontal bar graphs display data in a horizontal format. It's a graph of horizontally drawn bars. The vertical axis displays the data types, while the horizontal axis displays the data values. It is primarily used for bigrams and trigrams analysis.

KDE plot: The kernel density estimate (KDE) is a way of visualizing the distribution of visuals in a database, like a histogram. KDE represents data using a continuous curve of the magnitude of one or more dimensions. KDE plot is a non-

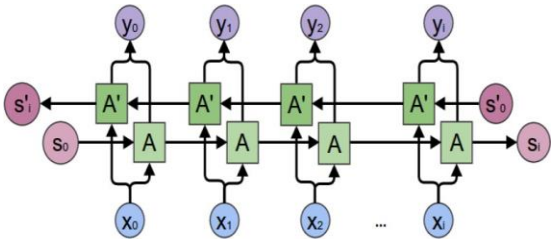
parametric way of estimation technique in statics

Strip plot: The entire usable area is split into a horizontal strip and b vertical strips in a strip plot. Each row is assigned one level of factor A, and each column is assigned one level of factor B.

Classification

This module deals with the classification. The word-embeddings obtained as an output from the previous step are divided into a test-set and a training set. The training data is used for training the classifier. We will start by studying traditional classifiers like Naive Bayes classifier, Support Vector Machines[SVM] and Logistic Regression. Then proceed with neural networks like DNN, LSTM and BiLSTM as classifiers with sigmoid activation function. Sigmoid is used here because the value should be a probability between 0 and 1 in order to predict the metaphoricity of the sentence.

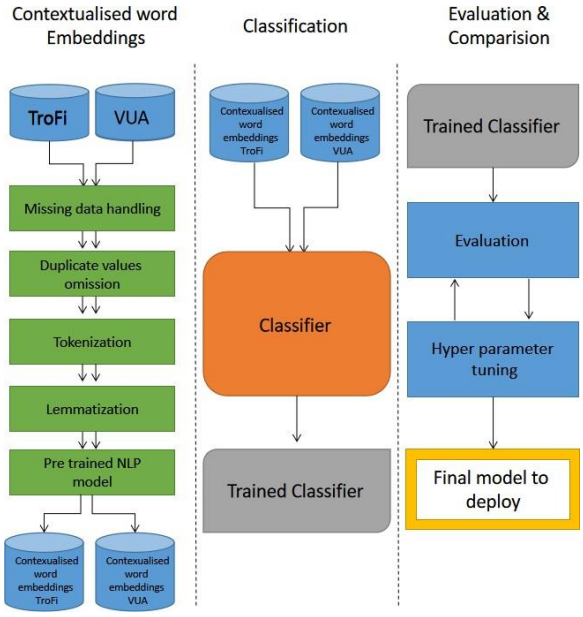
Using Bi-directional LSTM shown in figure 3.1, inputs will be run in two ways, one from previous to future and one from future to previous. This will help us in preserving information of both past and future.



Bi LSTM Architecture obtained from Swarnkar and Singh.

Architecture Diagram

Referring to Figure 3.2 We use pre-trained state-of-the-art NLP model (Devlin et al. (2018) BERT / Zhilin Yang (2019) XLNet) to get the contextualized word embedding and then pass the same to a different classifiers like the Dense Neural Network, LSTM or a two-layer Bi-LSTM (Shen et al. (2019) Bi-LSTM) neural network architecture. Later, the classifiers are evaluated and the hyper parameters are tuned and the model which yielded best results is finally deployed.



Architecture Diagram

Data Preprocessing

So the required modules are imported. We installed the Python transformer library, which includes a PyTorch implementation of BERT and several pretrained BERT models. For the implementation, we have used the TroFi and VU Amsterdam Datasets .The duplicates sentences were removed. Labels were manually added for simplifying the preprocessing and the Visualizations. Necessary visualizations were made .The stop words are removed and Lemmatization was made on the sentences.

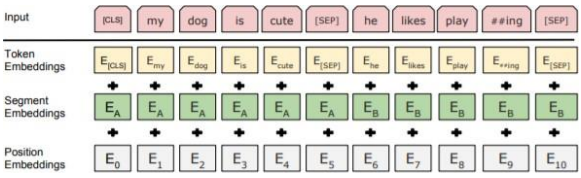
Pretrained Models

pretrained models are those that are already have been trained to solve similar problem. Instead of starting from scratch to solve a similar problem, you start with a model that has already been trained on another problem. The

following are some few famous pre- trained models that have been tested out.

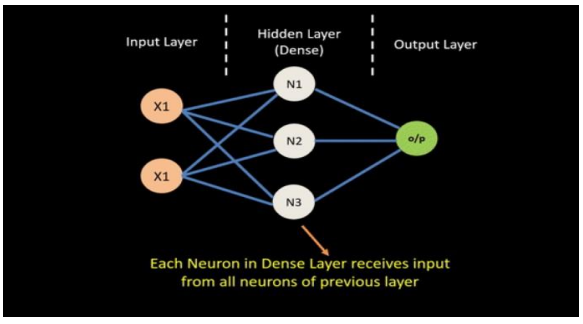
BERT

The innovation of BERT is transformer bi-directional training, a common attention model, language model . Bert is different to older works, which sees a text sentence all the way from left to the right or either way from right to left training. findings proved us that a bidirectionally trained language model will have much better sense of language back- ground and go in a single direction language model. The researchers describe a new technique called Masked LM (MLM) that allows bidirectional training in previously impossible models. BERT uses Transformer which is an attention grabbing method which studies the relation of status between the words or subworlds in a text. Transformer includes Two separate mechanisms in its final form - an encoder that reads the text input and the decoder that drugs edition for the task. Since the purpose of BERT is to create a language model, only enc. Google defines Transformer detailed works in per.



BERT Architecture obtained from Devlin et al

Dense Neural Network



A dense network is one in which each node's number of connections approaches the maximum number of nodes. Each node is connected to nearly every other node. A fully wired network is one on which every node is connected to every other node in the same way. The dense layer is an ordinary deep-connected neural network layer. The dense layer performs the following operations on the input and returns the result. Syntax: output = activation (point (input, core) + offset), where input represents input, kernel represents weighing data, point represents the numerical product of all inputs and their corresponding weights, and offset represents deviation. As a value used to optimize the model in machine learning, activation represents an activation function. The names of dense layers indicate that all these layers are completely interconnected by the neurons present in the network layer. Each of the neuron from one layer gets input from all of the neurons present in the last layer, so they are tightly connected to each other. to explain in other way, the tight layer is a severely connected layer, that is, all of the neurons in a single layer. all of These layers are connected to the neurons in the upcoming layer. The tightly coupled layer provides training attributes for all attribute combinations from the previous layer. The repeated fields are small. The dense layer of the neural network maps each of the neuron in one of the layer to each of the neuron in the next coming layer. This allows the potential function to be as close as possible within a

```
model = Sequential()

model.add(Dense(1000, input_shape=(30,), activation="relu"))
model.add(BatchNormalization(axis=-1))

model.add(Dense(256, activation="relu"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.25))

model.add(Dense(256, activation="relu"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.5))

model.add(Dense(128, activation="relu"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.25))

model.add(Dense(10, activation="relu"))
model.add(BatchNormalization(axis=-1))
model.add(Dropout(0.25))

model.add(Dense(1, activation="sigmoid"))
```

given slice width. This also means that many parameters must be set, so the training of very wide and very dense networks requires a lot of calculations.

DNN classifier model

LSTM

```
#LSTM layer arcitecture hyperparameters
n_lstm = 20
drop_lstm =0.5

model1 = Sequential()
model1.add(Embedding(30000, 64, input_length=30))

model1.add(LSTM(n_lstm, dropout=drop_lstm, return_sequences=True))
model1.add(LSTM(n_lstm, dropout=drop_lstm, return_sequences=True))
model1.add(Flatten())

model1.add(Dense(24, activation='relu'))
model1.add(Flatten())
model1.add(Dense(1, activation='sigmoid'))

# model1.add(Dense(1, activation='softmax'))
```

LSTM expanded as Long short term memory which is an artificial neural network architecture. LSTM has feedback relations, unlike normal feed-forward neural net- works. It can handle not just single data points, but also whole data sequences. Long- term and short-term memory networks (LSTM) which are kind of repetitive neural net- work which is efficient of learning the dependency of order in the sequence prediction problems. It is also used in complicated problems such as machine translation as well as speech recognition and even more. It can process entire data streams such as voice or video. A common Long-term and short-term memory networks(LSTM) component consists of cell an entry door an exiting door and also a forgetting door. This cell (the memory part of the LSTM unit) generally stores values at any time interval and on all of the gates. controls the flow of information in as well as out of the cell. Long delays with certain problems are overcome with LSTM, where buss noise, dispense representations and a continuous values are also processed. With the help of LSTM, there is no need for us to maintain a particular number of states in the cell. proceed is required in the Hidden Markov Model. LSTM generally offer a wide range of parameters like learning rates, entry and exit biases.

RESULTS AND DISCUSSION

The datasets were pre-processed and passed through the pretrained BERT model and the the contextualizes word embeddings were successfully generated. These contextualized word embeddings is then passed through a DLL, LSTM and Bi-LSTM neural network classifier respectively and the metaphoricity is calculated, evaluated and compared with the former models. The following are the results obtained.

VUA Dataset	
Model	Accuracy
BERT	0.71
XLNet	0.73

VUA Dataset.

TroFi Dataset	
Model	Accuracy
BERT	0.71
XLNet	0.75

Trofi Dataset Model

These are the results yielded the pretrained models for both datasets VUA dataset and TroFi datasets. The accuracy is compared between the BERT binary classifier and the XLNet binary classifier. These are, in other words the in-built classifiers of the pretrained models. As we can see, the XLNet binary classifier has performed relatively better than the BERT binary classifier.

VUA Dataset		
Algorithm	**Precision**	**Recall**
DNN	0.75	0.88
LSTM	0.78	0.80
Bi-LSTM	0.82	0.85

These are the results yielded by classifiers for both datasets VUA dataset and TroFi dataset. As

it is seen, the Bi-Lstm classifier coupled with the BERT pretrained model yields the maximum performance compared to the other classifier. This pattern can be observed in both the datasets. This is most likely because Bi-LSTM classifier is able capture long-range relationships between words in the same sentence.

CONCLUSION

Model was created successfully that could predict the metaphor city of a sentence. Contextualized embeddings were created using the BERT pretrained language model and the same is later passed to Classifier which is based on NN that has a sigmoid layer that outputs a likelihood based on both local and global contextual knowledge. Performances of different classifiers were compared namely the DNN, LSTM and Bi-LSTM when paired up with the BERT pretrained model. It's observed that the Bi-LSTM classifier is able capture long-range relationships between words in the same sentence which may reveal the presence of metaphors and has outperformed the former classifiers.

FUTURE ENHANCEMENT

The model is now fine tuned with respect to our use-case that is metaphor detection. The performance of the model can further be improved if any new models that have been pre-trained to the highest level can be implemented. In addition, since classifiers play an important role in the performance, the accuracy can be pushed further by implementing any new state of the art classifiers. Further innovation is possible when this model is fine tuned with a different language metaphoric dataset. Already previous researchers have implemented the model to a Chinese dataset. The idea of implementation of native language dataset over the model, native language could be Hindi, Telugu, Tamil, Malayalam and many others. Also further innovation is possible by translating the English language to an intermediate language which will later be translated to native language while keeping the metaphoric meaning intact. There are many good translators in the past like google translate but the key challenge would be to keep the metaphoric meaning of the sentence intact.

REFERENCES

1. Birke, J. and Sarkar, A. (2006). "A clustering approach for nearly unsupervised recognition of nonliteral language." *11th Conference of the European Chapter of the As- sociation for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

2. Brun, O., Yin, Y., and Gelenbe, E. (2018). "Deep learning with dense random neural network for detecting attacks against iot-connected home environments." *Procedia Computer Science*, 134, 458–463.

3. Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). "Bert: Pre- training of deep bidirectional transformers for language understanding. cite arxiv:1810.04805Comment: 13 pages.

4. Hall Maudslay, R., Pimentel, T., Cotterell, R., and Teufel, S. (2020). "Metaphor de- tection using context and concreteness." *Proceedings of the Second Workshop on Figurative Language Processing*, Online. Association for Computational Linguistics, 221–226.

5. Hochreiter, S. and Schmidhuber, J. (1997). “Long short-term memory.” *Neural Computation*, 9(8), 1735–1780.

6. Jerry Liu, Nathan O’Hara, A. R. R. D. C. R. (2020). “Metaphor detection using contextual word embeddings from transformers(2020).” *Association for Computational Linguistics*, Duke University. Association for Computational Linguistics, 250–255.

7. Pérez-Sobrino, P. (2014). “Gerard j. steen, aletta g. dorst, j. berenike herrmann, anna kaal, tina krennmayr trijntje pasma (2010). a method for linguistic metaphor identification: From mip to mipvu..” *Metaphor and the Social World*, 4.

8. Shen, M., Zhang, X., and Li, X. (2019). “Attentional bi-directional lstm for semantic attribute prediction..” *ICVIP*. ACM, 217–221.

9. Shutova, E., Sun, L., and Korhonen, A. (2010). “Metaphor identification using verb and noun clustering.” Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), Beijing, China. Coling 2010 Organizing Committee, 1002–1010.

10. Stemle, E. and Onysko, A. (2018). “Using language learner data for metaphor detection.” Proceedings of the Workshop on Figurative Language Processing, New Orleans, Louisiana. Association for Computational Linguistics, 133–138.

11. Sun, Y., Wang, S., Li, Y.-K., Feng, S., Tian, H., Wu, H., and Wang, H. (2020). “Ernie 2.0: A continual pre-training framework for language understanding..” AAAI. AAAI Press, 8968–8975.

12. Swarnkar, K. and Singh, A. K. (2018). “Di-LSTM contrast : A deep neural network for metaphor detection.” Proceedings of the Workshop on Figurative Language Processing, New Orleans, Louisiana. Association for Computational Linguistics, 115–120.

13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). “Attention is all you need.” Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H.Wallach, R.Fergus, S.Vishwanathan, and R. Garnett, eds., Vol. 30. Curran Associates, Inc.

14. Wu, C., Wu, F., Chen, Y., Wu, S., Yuan, Z., and Huang, Y. (2018). “Neural metaphor detecting with CNN-LSTM model.” Proceedings of the Workshop on Figurative Language Processing, New Orleans, Louisiana. Association for Computational Linguistics 110–114.

15. Yu Sun, Shuohuan Wang, Y. L. S. F. H. T. H. W. H. W. (2019). “Ernie 2.0: A continual pre-training framework for language understanding(2019).” Computation and Language (cs.CL). Computation and Language (cs.CL), 3–11.

16. Zhang, D., Lin, H., Liu, X., Zhang, H., and Zhang, S. (2019). “Combining the attention network and semantic representation for chinese verb metaphor identification.” IEEE Access, 7, 137103–137110.