Explorations on Meaning Map Methods

*Note: Sub-titles are not captured in Xplore and should not be used

1st Bedirhan Çelebi
Masters of AI (student)
Özyeğin University
İstanbul, Türkiye
bedirhan.celebi@ozu.edu.tr

***Abstract*—Machine learning algorithms require words to be represented as vectors. One hot encoding is a method to vectorize words but it results in a lot of sparse matrices and it requires a lot of memory. Also one hot encoded vectors do not represent the relationships between the words. There are many embedding methods to approach this problem. I am planning to use a language dictionary to embed words in a structure which is similar to term-term matrices and I will compare this method with the Simlex999 dataset. [4] (Github repo : https://github.com/sungurula/MeaningMaps.git)**

***Index Terms*—Meaning map, embedding, context, Cortex, Representation in the brain**

I. Introduction

Information is stored and processed in the brain with billions of neurons and trillions of connections. This structure has biological neurons as the base computing unit, complex chemical reactions for the algorithms and biological processes. Discovering the workings of mind is difficult from bottom up so what if we try to do it top down. We come up with a comprehensive hypothesis which is representing everything encountered in the world in the brain with connections between groups of neurons. So, when we see a car a group of neurons and their connections would light up to create the representation of the car.

However, we have many senses such as vision, scent, hearing and many other. Therefore, lighting up the neurons of the car would require lighting up the neurons that represent a color, or some sound or smell, some vision neurons to remember or represent the features of the car.

I thought of creating such a structure by using big tensors with each matrix in the tensor dedicated to a type of sense. A matrix in a tensor represents the connections of a group of neurons who are responsible for remembering or representing a thing in that sense. So a sense matrix can inspire, excite, inhibit, associate with other matrices resulting in a structure where like in the brain we could be reminded of a "strawberry taste" when we see pink, on our car or confuse butter for ice cream and tastes accordingly. To test this approach, I thought of using human language as a sense or space of meanings in which we can create nodes that connect to each other in a context collaborating to represent a thing in the real world.

Language was studied intensely for a long time in the AI community and there are many good models that generates [9] [5], understands, parses or translates languages. Putting language on the focus might be a practical thing and Chat GPT is a great example, Generative Pretrained Transformers are indeed talented language creators [1], [10], [12], however, the structure which creates language in human brains sounds more interesting to me. One needs to learn the natural language processing methods however to really get a good sense of the human language to analyze and observe the success of such innovative paradigms.

Natural Language Processing algorithms require words to be represented as vectors. These vectors are of N X 1 dimensions. They are asked to represent the context and meaning of words. There are some methods developed to do it such as word2vec [2], one hot encoding , glove [7], term term matrices or some other embedding methods.

I am proposing that instead of having a model learn from a text corpus to learn word embedding, we can create a meaning map from a dictionary which will technically be a term term matrix with some special adaptations made to function in a more adaptive way. In this paper the performance of the meaning maps embedding algorithm will be put to test against Simlex999 [4] dataset. Out of the scope of this paper, meaning maps can be trained on a corpus after being trained on a dictionary to make the embedding dense rather than very sparse. We can cut out the part of the huge matrix in which updates have happened a lot to the word vector and we can create dense embeddings from this matrix.

A. Data

According to the aim of this paper we need to have a dictionary type data that has the form described below. "word : definition".

- The Online Plain Text English Dictionary (OPTED) I use this dictionary from kaggle. Following is the link to the dictionary howeveri authors and many other details have not been represented.(Appendix A)
- Simlex999 [4] dataset to evaluate the metrics.

II. Related Work

Categorical Word Representation

In order to enable computers to process human language, it is necessary to convert words into mathematical representations. Categorical word representation seeks to represent words in vector forms.

One-Hot Encoding

The simplest method is one-hot encoding, which represents a word as a vector with zeros in its positions except for one position that has "1" as a value.

$Word1 = [0, 0, 0, 0, 1]$

Term Term matrices Representation

This method has a similar embedding algorithm as the meaning maps.The difference is that term-term matrix has a window which fills a table of frequencies. Meaning maps take things into more semantic frames where a window does not exist but rather examples and definitions exist.

There are other important method of embeddings as well. However, we will not be utilizing them in this paper. Continuous Word Representation (Non-Contextual Embeddings)

Word2Vec [2]

Word2Vec is a method that aims to vectorize words while keeping words with similar contexts closer to each other. It achieves this by employing two shallow networks: CBOW (Continuous Bag-of-Words) [11] and Skip-gram models [6]. These models help capture the relationships and context of words within a given text.

III. Method

Meaning maps have a curse like the term-term matrices have which is sparsity and huge matrix size. However, for our purpose of understanding the brain which uses many biological analog sparse matrices, it is a good concept to experiment with. I think the fastest way we create a general meaning map is to use a language dictionary. "word:definition". Dictionaries have the previous structure. They usually have more than one meaning or stronger and weaker meanings in the language. In these structures we create our meaning map by iteratively going through the "word:definition" pairs. For each word, we create nodes that represent the word and definition words if they have not already been created and make connections to other words which are represented as other nodes. We go through the dictionary and when two words are used in the same definition or key we create a connection between. Let us see a simple example:

"Spoon" : " a type of utensil used for mainly drinking soup." This sentence would crate a 7 X 7 matrix with each of the values being equal to one. As the new "word:"definition" pairs added the matrix will increase in size exponentially bu then the speed of increase in the size of matrix will slow down because all the words has actually been encountered before so they will not be recreated and added to the meaning map matrix.

More sophisticated methods could be used to give different weights to the connections. For example, first meaning and second meaning words might have different connection weights with each other resulting in a better embedding and representation of the connection.

All the nodes and connections are represented with a huge sparse matrix.

Each of the words in the dictionary the key and definition words have given a connection. Finally we end up with a structure where there is a huge matrix, each column and each row represents a word and the values of the matrix are how many times those words have been used in a definition before.

After getting that matrix, Simlex999 [4] dataset will be used to test the semantic power of the meaning map. Simlex999 [4] dataset has many word pairs which were given points by humans describing the similarity of the words to each other semantically.

So word : "A" , has many pairs. For example, "A,B" : 5 , "A,C":1, "A,D" : 10 When we order them according to their similarity grades we will have the ordering, "A,D" : 10 "A,B" : 5 "A,C" : 1

To test the meaning map against this algorithm cosine similarity [8] is used. We are going to get the word pair vectors from the meaning map and get their cosine similarity to get their similarity scores and then order them like we did the SimLex999 and see if it will give us a similar ordering.

$$\frac{\vec{a}.\vec{b}}{|\vec{a}| * |\vec{b}|} \tag{1}$$

IV. Experiment

In this part we use the OPTED English dictionary. Algorithm starts by initializing a "sözlük" meaning a dictionary data type. This data type holds the words and gives them a name that points to the corresponding word in the meaning map matrix. With each word added we concatenate that value into the main meaning map matrix.

Once we go through the whole dictionary now we are ready to compare it against some of the word pairs in the SimLex999 dataset.

This is an example comparison. "old , X" word pairs are tested on their similarity. In this case Meaning map ordered the words in accordance with the Simlex 999 dataset. "MM" column is the Meaning Map assessment.

word1	word2	Simlex999	MM
old	new	1.58	0.49
old	fresh	0.87	0.34

There are some other examples as well. When we look at this example we see that the Meaning Map gets confused a bit. Here in this example, "hard, X" and "hard, simple" pairs were graded and ordered wrongly. Other results are correct.

word1 word2 SimLex999 MM
2 hard difficult 8.77 0.778565
4 hard easy 0.95 0.71132
14 hard simple 1.38 0.833276
19 hard tough 8.05 0.752856
98 hard dense 5.9 0.726261

V. Evaluations

A method is needed to objectively evaluate the success of the meaning map. Which could be easily made by counting the correct orderings. Any wrong placement of an object would mean a wrong ordering and any correct would mean a correct ordering. I have designed that algorithm and run it. This algorithm works by grabbing some word pairs with the same first word. Then it orders those names first by "SimLex999" column and then by the "MM" column.

After the algorithm goes through the process when there is a wrong answer encountered it removes that instance from the list that we are ordering the pairs in and restarts the loop. Finally we end up with the overlapping ordered list.This list shows us the values that have been correctly ordered. Number of pairs removed is the number of wrong orderings. In the end we end up with the values of the same ordering. The bigger the final list the better since it would mean fewer errors.

Here are the results for different number of words that are used to construct the meaning map and their orderings compared with the SimLex999 dataset. Accuracy shows us the success of the Meaning map after trained with one dictionary.

word count	correct	wrong	accuracy
2000 word	407	132	0.75
5000 word	388	134	0.74
10000 word	372	142	0.72
20000 word	367	143	0.71

From the table we can see that the meaning map can represent around 71 percent of the SimLex999 dataset semantic relations by using a language dictionary and a novel frequency based algorithm. I applied min-max scaling to the vectors however, it does not make a difference on cosine similarity measure. I also get rid of usual stopwords of english which does not have a meaning by themselves and only mean something with other words such as " was", "were", "with","on","in" etc.

VI. Discussion

Embedding is an interesting topic in NLP and many new methods could be invented to create new approaches toward creating a semantically coherent representation of sentences and words. [3]

By using a meaning map I was creating a bigger hypothesis. Rather then using only on NLP I am planning to widen it to create a bigger system of memory. In that approach I think a meaning map needs to be more different than the Term Term frequency matrix. In that manner we need to make meaning map a more dense embedding. To elaborate, the meaning map in this paper can be trained using a dictionary is a general meaning map. To make it denser and make topic specific such as "medicine,law or physisc" we need to use other corpus-es. By trying the algorithm on other corpus-es we will cut out the more frequently updated values and create an embedding specific for that topic. This indicates that from the main meaning map we create different concepts that have different dense embeddings for the word. Those newly adapted dense embeddings will be the representetives of the neuron groups in the brain.

However, for this paper it is not possible to include everything. Instead I tested the embedding capacity of the meaning map with the Simlex999 dataset. I think if I would have used more data then the meaning map would perform better since there are problems with the definitions in the dictionary such as defining a word with its antonym makes a strong relation between the two word vectors but it is not represented in the SimLex999. Also, making meaning map more professional by training with other corpuses and making connections with those smaller meaning maps might be of great help to counter this issue since more related words will be close each other rather than words and their antonyms.

VII. Conclusion

To conclude, NLP is an open research area. It is open in a way that we do not know how the biological neurons and biological structures produce language. So I take a neurological viewpoint and make a crude assumption about how the memory works. I call those memory structures meaning maps In this paper I demonstrated the potential capacity of a first degree meaning map which uses only a language dictionary to train and embed words. It can make semantic connections with 71% accuracy when we take the SimLex999 dataset as reference. Main problem of this project was the sufficiency of the dictionary. It might have problems with the number and length of the definitions.

A second degree meaning map would require the update and sampling from a meaning map that has been trained on concept specific corpus. By cutting out the most updated values we have a concept vector which could be used for the specific topic.

A third degree meaning map would require more than one second degree meaning maps connecting to each other on different senses. For example, to embed the text of a movie you need to also embed it with the visual, color or feature information of the movie . In the end, by using around "seven" second degree meaning maps connected and creating a one third degree meaning map ,this structure hypothetically would approximate human conceptualization.

VIII. Appendix

Here are some important links that lead to the code and the data used for this project.

- Link for OPTED dictionary : https://www.kaggle.com/datasets/dfydata/the-online-plain-text-english-dictionary-opted
- Link for the code and data used for the project: https://github.com/sungurula/MeaningMaps.git
- Link for the SimLex999 dataset : https://fh295.github.io/simlex.html

IX. References

References

[1] Martin R. Chavez, Thomas S. Butler, Patricia Rekawek, Hye Heo, and Wendy L. Kinzler. Chat generative pre-trained transformer: why we should embrace this technology. *American Journal of Obstetrics and Gynecology*, 228(6):706–711, 2023.

[2] KENNETH WARD CHURCH. Word2vec. *Natural Language Engineering*, 23(1):155–162, 2017.

[3] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method, 2014.

[4] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation, 2014.

[5] Touseef Iqbal and Shaima Qureshi. The survey: Text generation models in deep learning. *Journal of King Saud University - Computer and Information Sciences*, 34(6, Part A):2515–2528, 2022.

[6] Angeliki Lazaridou, Nghia The Pham, and Marco Baroni. Combining language and vision with a multimodal skip-gram model, 2015.

[7] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[8] Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi. Semantic cosine similarity. In *The 7th international student conference on advanced science and technology ICAST*, volume 4, page 1, 2012.

[9] Sivasurya Santhanam. Context based text-generation using lstm networks, 2020.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[11] Qi Wang, Jungang Xu, Hong Chen, and Ben He. Two improved continuous bag-of-word models. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2851–2856, 2017.

[12] Junping Zhang, Jian Pu, Jianru Xue, Ming Yang, Xin Xu, Xiao Wang, and Fei-Yue Wang. Hivegpt: Human-machine-augmented intelligent vehicles with generative pre-trained transformer. *IEEE Transactions on Intelligent Vehicles*, 8(3):2027–2033, 2023.