Comparing Interpretability of Biologically Inspired and Traditional Word Embeddings

Tristan von Busch[1] and Ferdinand Spurny[2]

[1]Department of Informatics, University of Hamburg
[2]Department of Informatics, University of Hamburg

January 2022

Abstract

To build state-of-the-art natural language processing systems, computers need sophisticated representation of the basic building blocks of language - words. Several so-called word embedding techniques exist that have interesting properties in their representations of words. Many of these approaches have been shown to increase the performance of machine learning models on different natural language processing tasks. One aspect of word embeddings that is often neglected is their interpretability and explainability. In this paper we take a closer look at a recently proposed biologically inspired word embedding technique called FlyVec and systematically evaluate its capability to encode context into specific neurons. The FlyVec model is based on the simple neurological architecture of the mushroom body in fruit flies. One aspect of this model is that it can produce binary and therefore sparse word embeddings of high quality by modeling strong inhibitory signals between layers of neurons. We examined if the biological model has inherent advantages over artificial word embeddings in terms of its interpretability and explainability. Our results show that the dense FlyVec word embeddings were indeed able to outperform other common techniques in terms of an interpretability score.

1 Introduction

In our day-to-day lives, we already have encounters with a variety of different systems and applications that use natural language processing and vector space representations of words. From machine translation to question answering, a lot of different use cases and tasks are using approaches based on machine learning and neural networks. What all of these models and applications have in common is that they were trained and built on the foundation of natural language.

We have come a long way from methods like one-hot and tf-idf encoding [2] to sophisticated distributed word embeddings like word2vec [4], GloVe [6] and BERT [1] that have proven to improve performance of language models on downstream tasks and capture semantic relationships. However, methods like these, that are based on neural networks, still require large amounts of storage space and training resources to achieve these results. Brains stand in contrast to this having been shaped by evolution to survive in environments containing sparse resources and ample competition. Therefore more efficient model architectures that are closer to real biological neural systems have the potential to reduce costs and improve efficiency. This makes them an interesting target to eventually contain mechanisms applicable to currently researched data processing tasks. The mushroom body in the brain of the fruit fly has been extensively studied and is one such target, as it contains a neural communication pattern based on controlled, strong inhibition [3].

In this paper we want to further investigate a biologically inspired word embedding technique that is based on the neurophysiology of the fruit fly and was proposed by Liang et al.[3]. Specifically, we take a look at the interpretability of the word embeddings in regard to encoding of context and semantic as well as syntactic relationships.

Explainable and interpretable systems are important for more complex tasks that need to have a high fault tolerance and require a lot of confidence in the system. These more complex tasks may include inferring facts about relationships, making predictions based on new data, or more generally performing new kinds of reasoning. A largely unexplainable system can significantly decreases confidence even if certain performance benchmarks seem promising.

2 Background

Word embeddings are a method for representing words in vector spaces [2]. Because they map words and sentences (e.g. strings) to vectors of numbers, they can be used as inputs to various artificial neural network architectures. These word representations can increase performance of language models on downstream tasks [1] and also have several useful properties like for example the ability to construct word analogies by simple vector arithmetic (e.g. $\vec{king} - \vec{man} + \vec{woman} \approx \vec{queen}$) that was outlined by Mikolov et al. [5]. The following section is going to provide an overview of some of the work in the domain of biologically inspired word embeddings and briefly touch on the three commonly used embedding techniques word2vec [4], GloVe [6] and BERT [1] which do not have a biological inspiration.

2.1 Biologically Inspired Word Embeddings

Many of the more prominent and popular methods for word representations and word embeddings, that are outlined in more detail in the next sections, are not based on biologically inspired concepts. However, some approaches for the construction of word embeddings exist which are based on concepts from biology. The FlyVec model, which will be the main model we will examine in this paper, was proposed by Liang et al. [3] and the approach is based on the neural architecture of the mushroom body of the fruit fly.
Another technique to obtain word embeddings as spare binary representations was proposed by Yuwei et al. in [10]. For this they used the biologically-inspired concepts of neuron coding and spiking neural networks.

2.2 word2vec

The original paper by Mikolov et al. [4] presented two different methods for inferring meaning from vector word representations. Both of these methods - continuous bag-of-words (CBOW) and Skip-gram - are depicted in the original visualization from the paper in Figure 1. While the continuous bag-of-words (CBOW) model infers the most likely (i.e. least distant) word given a selection of k context words, the Skip-gram model takes one word and outputs the most likely collection of k context words [4]. Both take a vector from a V-dimensional vector space as input with V as the size of the vocabulary [4]. The entries represent words of the vocabulary in one-hot encoded form in the Skip-gram model and k-hot encoded form in the CBOW model. In either case a single hidden layer is used to transform the input to latent space [4]. Training is done on large corpora of text in an unsupervised way through back propagation. Weights are adjusted according to whether words could correctly be inferred from the input vector [4].

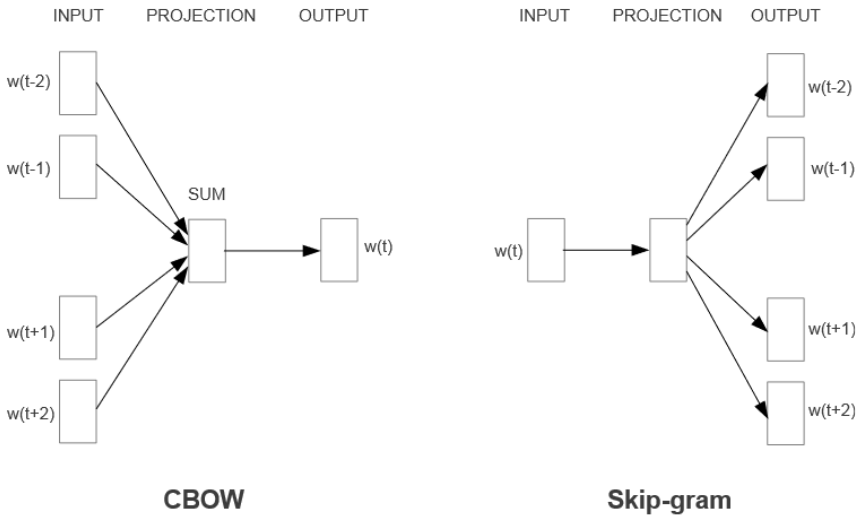

Figure 1: architecture of CBOW and Skip-gram models [4]

2.3 GloVe

GloVe is a model introduced by Pennington et al. [6] based on the notion that similarity between words can be captured from their shared occurrence in text and converted to vector representations which allows distance measurements between these word vectors. To quantify these relations a symmetric co-occurrence matrix X is constructed which represents the probability of joint appearance $X_{i,j}$ for each pair of words (i, j) [6]. This is done by first constructing a raw co-occurrence count matrix, in which each context is then normalized by dividing it by the sum of all contexts in its row (i.e. all the contexts of a given word) [6].
The authors use the appearances of the words "ice" and "steam" to demonstrate how information is instilled in the matrix (see Figure 2). The probability $P(k|ice)$ denotes how often a word k like for example "solid" appears in the context of "ice" relative to all other possible context words. By dividing the entries one can obtain the relative appearance ratio of a word in different contexts [6].

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figure 2: Probability of co-occurrences of words k given "ice" and "steam" as well as their relative relations to one another [6]

2.4 BERT

The Bidirectional Encoder Representations from Transformers, or BERT for short, are a more recent approach to obtain contextualized word embeddings that was introduced in 2018 by Devlin et al. [1]. The network architecture of the BERT model is very different from previous methods to obtain word embeddings like word2vec [4] and GloVe [6]. BERT uses a multi-layer bidirectional transformer as introduced by Vaswani et al. [9] that is based on an encoder-decoder architecture that solely relies on a self-attention mechanism and has no recurrent connections like many previous approaches. BERT has shown to achieve state-of-the-art results on a variety of downstream tasks [1]. The training process for BERT happens in the two steps called pre-training and fine-tuning. The pre-training is performed on large corpora of texts as it is the case for many other language models. However, during the fine-tuning the inputs and outputs for specific tasks are used and the model is trained further on the basis of previous pre-training [1]. This way the model can be adjusted relatively efficiently to specific downstream tasks.

3 FlyVec

The FlyVec model that was proposed by Liang et al. [3] is a biologically inspired word embedding technique which we want to examine and evaluate more closely especially with respect to its interpretability. Based on the neurophysiology of the mushroom body in fruit flies, it takes a significantly different approach to learn word embeddings when compared to methods like word2vec, GloVe or BERT. The part of the mushroom body of the fruit fly that is modeled by FlyVec consists of three types of different neurons - the projection neurons (PN), the Kenyon cells (KC), and the anterior paired lateral neuron (APL) [3]. The projection neurons compose the input layer of the neural network and pass a diverse set of sensory stimuli to the layer of Kenyon cells. These inputs are modulated by synaptic weights which are periodically updated to train the model. The activations of the Kenyon cells are in turn influenced by the anterior paired lateral neuron [3]. The connections and architecture of these types of neurons are visualized in Figure 3.

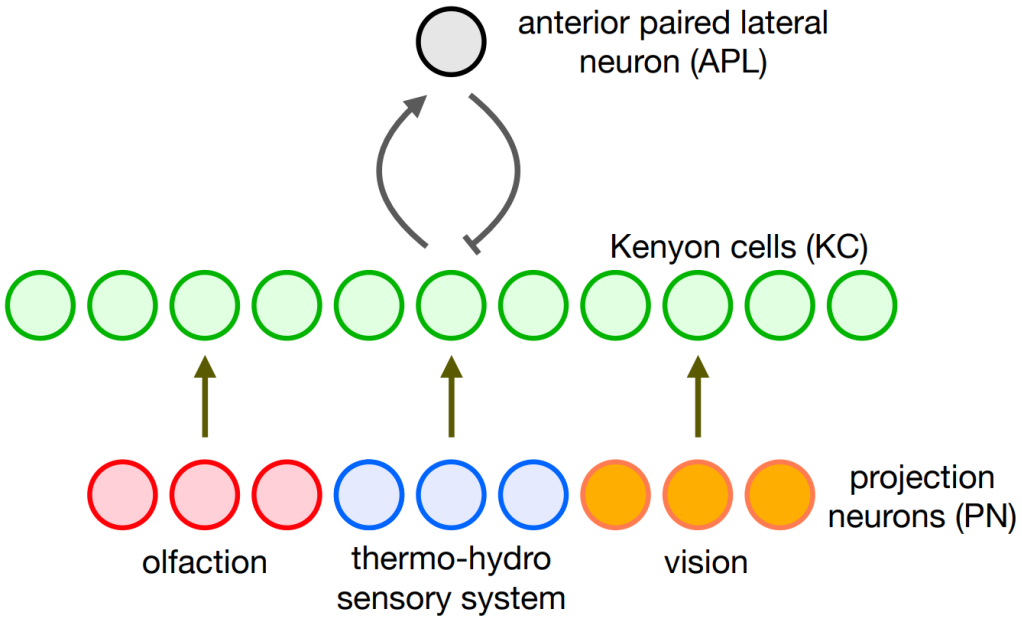

Figure 3: Schematic of fruit fly neural architecture [3]

The fruit fly network is not a deep network like it is the case for several other word embedding techniques. The network is also not trained by the means of backpropagation but uses the minimization of an energy function [7] as its learning rule to update the synaptic weights between projections neurons (PN) and Kenyon cells (KC) [3].

The inputs to the network are sparsely encoded into a vector V of length $2N_{voc}$

and forwarded from the projection neurons to the Kenyon cells where N_{voc} is the size of the vocabulary that is used for training. The first N_{voc} elements of V are referred to as the context block and the last N_{voc} elements are called the target block [3]. The target block is always populated and contains the one-hot encoded target word. If context-dependent embeddings are learned in contrast to static ones, the context block is used as well and contains the surrounding of the target words (context) encoded as a bag of words[3]. For static embeddings the context block is always empty and only contains zeroes. All the inputs from the training corpus are now generated by iterating over all the words in the corpus and encoding them into a vector of length $2N_{voc}$ as described [3]. For context-dependent embeddings the algorithm considers n-grams with the target words in the middle and encodes the $n-1$ context words into the context block of the input vector. In addition to this set of encoded input vectors, the training algorithm also needs a real-valued vector of length N_{voc} that contains the occurrence probability of each word in the corpus vocabulary [3].

The training of the network is supposed to adjust the synaptic weights in a way that similar input vectors activate the same Kenyon cells in the end. For this Liang et al. [3] rely on the assumption that single elements in the input can vary and represent synonyms but the overall structure and relationships between words are preserved. The learning rule which achieves the minimization of the energy function is given in the following equation [3]:

$$\triangle W_{\mu i} = \epsilon g\Big[\sum_j W_{\mu j} v_j^A\Big]\Big[\frac{v_i^A}{p_i} - \Big(\sum_j W_{\mu j}\frac{v_j^A}{p_j}\Big)W_{\mu i}\Big] \tag{1}$$

W is the matrix of synaptic weights between the input layer of projection neurons (PN) and the layer of Kenyon cells (KC). $W_{\mu i}$ denotes the weight of the connection between the projection neuron i and the Kenyon cell μ. Therefore $\triangle W_{\mu i}$ describes the required update of the weight of that connection after applying the learning rule. ϵ is the learning rate of the algorithm to control the updates on the synaptic weights. v^A is the input vector of length $2N_{voc}$ and p denotes the probability of a specific word to occur in the training data [3].

Once the synaptic weights between the projection neuron layer and the Kenyon cells are trained, the network can be used to obtain dense word embeddings. The biological model of the mushroom body however also contains the anterior paired lateral neuron (APL). The anterior paired lateral neuron (APL) sends a strong inhibitory signal to the layer of Kenyon cells (KC) [3]. Mathematically this is modeled as so-called bio-hashing [7] where a fixed amount of Kenyon cells have their activation "dampened" to zero and the activations of the entire layer are effectively binarized. This means that the binary hash code representation of words contains k ones and $K-k$ zeroes where K is the number of Kenyon cells and k is referred to as the hash length [3].These sparse hash codes are then significantly more memory-friendly than their dense representations. In both cases the activa-

tions of the Kenyon cell layer yield the respective word embeddings. The number of Kenyon cells K therefore determines the dimensionality of the embedding space.

4 Evaluation

4.1 Interpretability Score

Evaluating the interpretability of machine learning models often requires a human factor because interpretability essentially refers to the ability of a model to make sense to humans. The issue with letting humans evaluate the interpretability of models is that it does not scale well and results are always subjective and therefore not necessarily reproducible.

To quantitatively evaluate the interpretability of word embeddings techniques Şenel et al. [8] came up with the idea of an interpretability score that can be computed for different word embeddings. For this they constructed and used a dataset called SEMCAT [1] which consists of several semantic categories (e.g. animals, sports, metals) that contain words which are associated with these individual categories. The interpretability of the word embedding techniques is then evaluated by performing a comparison of the different dimensions in the embedding space with the categories in the SEMCAT dataset to evaluate the models ability to represent semantic contexts in certain dimensions of the embedding space [8]. The algorithm to compute the interpretability score compares the set of words in each category of the SEMCAT dataset with the words that have the highest activations in each dimension of the embedding space [8]. The interpretability score of a model is therefore influenced by the ability of a single embedding dimension to reflect an entire category of the SEMCAT dataset.

This metric is also what we are looking for to evaluate the interpretability of the FlyVec model. We assume that the FlyVec model has the ability to encode contexts or semantic categories into single Kenyon cells (which are the embedding dimensions). The interpretability score metric allows us to examine how well the FlyVec model is able to do this and how it compares to other word embedding techniques. It is however important to note that the interpretability score is evaluated on the dense FlyVec embeddings because each dimension needs to be sorted prior to comparison with each category. The sparse, binary hash codes cannot be used for this without altering the algorithm to compute the interpretability score which is why we will evaluate the interpretability of only the dense embeddings.
With this interpretability score we can perform an initial quantitative evaluation of the FlyVec word embeddings with respect to their semantic interpretability. To put the obtained results into context, we also calculate the interpretability scores for other word embedding techniques like word2vec [4], GloVe [6] and more recent transformer-based embeddings like BERT [1].

[1] https://github.com/avaapm/SEMCATdataset2018

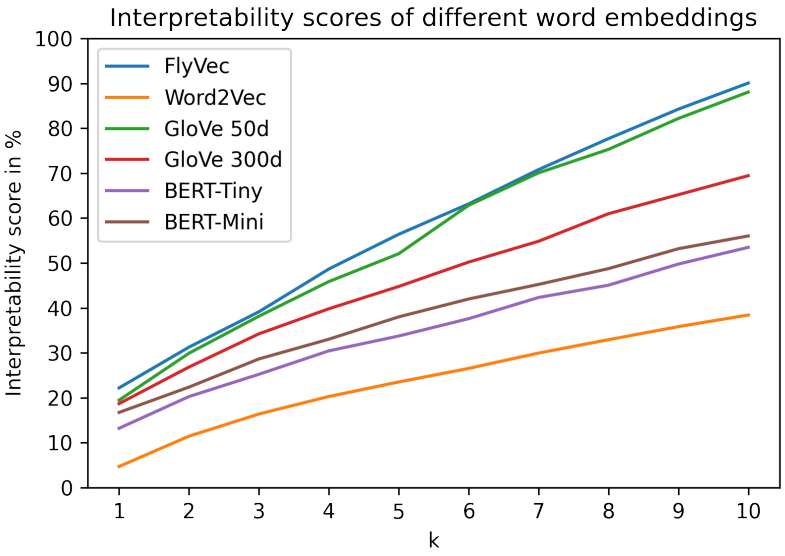

Figure 4: Interpretability scores of different word embeddings

In Figure 4 the interpretability scores for a variety of different word embeddings techniques including the FlyVec embeddings are shown. All of the interpretability scores were evaluated for the most common 20000 words in the vocabularies of the respective underlying models. k denotes the strictness of the requirements on the interpretability as described in [8]. $k = 1$ describes the strictest evaluation criterion.

The word embeddings obtained from the FlyVec model performed the best followed by GloVe, BERT and then word2vec. One interesting result is that the GloVe model that produced word embeddings with 50 dimensions performed better than the GloVe embeddings with 300 dimensions especially for greater values of k. Our assumption is that this is the case because 50 dimensions are closer to the amount of categories (110) that are included in the SEMCAT dataset. If these categories have to be distributed among more dimensions it could lead to a lower interpretability score because now a concept or category is no longer represented by a single but multiple dimensions in the embeddings. This stands in contrast to the much better results of the FlyVec model although it has even more dimensions in the embedding space (400). This indicates that the FlyVec model still encodes concepts and semantic categories into single Kenyon cells (dimensions in the embedding space) and does not distribute them across multiple dimensions. This ability of the FlyVec model to keep concepts localized in specific Kenyon cells results in a much higher interpretability score.

It is also noteworthy that even though the FlyVec embeddings with 400 dimensions performed the best overall they are only marginally better than the pretrained GloVe embeddings with 50 dimensions that have a much smaller memory footprint.

4.2 Semantic and Syntactic Relationships

One of the interesting properties of the original word2vec model my Mikolov et al. [4] was its ability to model analogies between words as simple arithmetic operations on the word embedding vectors. Because it is such an intuitive way to evaluate the quality of word embeddings, we would argue that this method also produces valid metrics to evaluate the interpretability of word embeddings. The often cited example [5] of $\vec{king} - \vec{man} + \vec{woman} = \vec{queen}$ is an equation that is correctly "solved" by word2vec models and is also intuitively understandable to a human. The abstract concept behind the computation $\vec{king} - \vec{man}$ would be something like $\vec{royalty}$ and the sum $\vec{royalty} + \vec{woman}$ should intuitively be very similar to $\vec{queen}$.

We can also look at other word embeddings techniques that were developed since word2vec and evaluate them on the same dataset of semantic and syntactic relationships that Mikolov et al. constructed for their evaluation of the CBOW and skip-gram models [4]. While our previous evaluation of the interpretability score mainly looked at the ability of individual embedding dimensions (or neurons in the case of FlyVec) to encode certain concepts, this evaluation on the semantic and syntactic relationship dataset will focus on a different aspect and property of word embedding techniques that is also important for interpretability and explainability of these models. As in the previous evaluation of the interpretability scores, we will compare the four architectures word2vec [4], GloVe [6], BERT [1] and the biologically inspired FlyVec [3] word embeddings.

To evaluate the ability of the different word embeddings to form semantic and syntactic relationships as well as model analogies, we used the semantic and syntactic relationships dataset[2] by Mikolov et al. [4]. Given a tuple of words that have a specific kind of relationship, the task was to correctly predict the word that has the same kind of relationship to another given word. The "prediction" of the word embeddings is in this case constructed by applying the same arithmetic operations outlined above and searching for the word closest to the resulting vector representation in the embedding space [4]. If the dataset for example contains the relationship between the words *deep* and *deeper* and the word embedding should "predict" the word that has the same kind of relationship to the word *high*, we would calculate $\vec{deeper} - \vec{deep} + \vec{high}$ and search for the closest embedding in the embedding space of the respective model. If the closest word in the embedding space is indeed *higher*, the "prediction" of the embedding was correct.
We used a limited vocabulary of the 20000 most common tokens in the training vocabulary of each model. This was done because only the 20000 most common embeddings were available for the pretrained FlyVec model. For each model we calculated the overall accuracy of the predictions for the entire semantic and syntactic relationship dataset. Because these restricted vocabularies would likely result in not all of the samples from the dataset having a representation in the embedding

[2]https://github.com/tmikolov/word2vec/blob/master/questions-words.txt

spaces, we only calculated the accuracy based on the samples for which each word was present in the embedding space. It is also important to note that for these experiments we used the sparse binary word embeddings obtained by the FlyVec model.

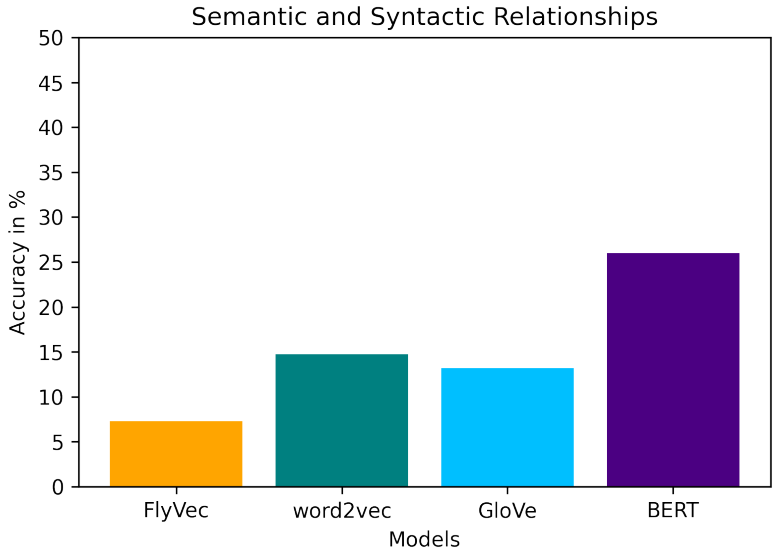

Figure 5: Accuracy on semantic and syntactic relationships

The results of our experiments are shown in Figure 5. word2vec and GloVe performed similarly and achieved an accuracy of roughly 15 percent. BERT as a more recent state-of-the-art model performed significantly better with around 25 percent accuracy. Although the FlyVec word embeddings performed the best overall on the interpretability score, with only 7.5 percent accuracy they showed the worst results out of all the tested models on the semantic and syntactic relationships. These results show that the modeling of semantic and syntactic relationships is not one of the strengths of the FlyVec model.

5 Conclusion

Liang et al. [3] already confirmed that the FlyVec embeddings can hold their own against popular word embeddings like word2vec [4], GloVe [6] and BERT [1] on a variety of different downstream tasks especially when comparing the sparse, binary embedding versions for each of the models. In this paper we evaluated the FlyVec model specifically on tasks related to model interpretability and explainability. We used the interpretability score for word embeddings that was introduced by Şenel et al. [8] to evaluate the interpretability of FlyVec in comparison to other word embedding techniques. Although it requires significantly less computational resources, the pretrained FlyVec model was able to outperform all other models in terms of the interpretability score of its word embeddings.
On the other hand, we also specifically examined the models ability to form semantic and syntactic relationships between different embeddings based on the dataset constructed and used by Mikolov et al. [4] for the evaluation of word2vec. The results we obtained from these experiments painted a different picture. On this metric the FlyVec model performed much worse then the other three tested word embeddings. The performance on the different metrics indicates that the biologically inspired FlyVec model has certain advantages and disadvantages in comparison to other word embedding techniques. However, both tasks target different aspects of interpretability. When looking at the goal behind the learning rule in the FlyVec model we see that it focuses on activating the same Kenyon cells (which are the embedding dimensions) for inputs that are similar from a semantic perspective. The interpretability score metric [8] basically evaluates this property to encode semantics into single neurons or embedding dimensions. As the results for the interpretability score show, the FlyVec model achieves this task better than other word embeddings that are not biologically inspired.
Overall, this suggests that even simple biological systems like the mushroom body of the fruit fly brain offer inherent advantages in terms of model interpretability and explainability as well as resource usage when compared to other state-of-the-art approaches to obtain word embeddings.

6 Outlook

The results of biologically inspired word embeddings in general and the FlyVec model specifically seem promising. The model was not only able to outperform traditional models on various natural language processing tasks [3] but also proves to be more interpretable than these models on certain metrics like the interpretability score. Although the FlyVec architecture and the mushroom body as its biological base are on paper very simple models compared to some other techniques to obtain word embeddings, they are nonetheless quite powerful. If even such simple biologically inspired models can achieve the outlined results, more complex neural architectures like the ones that are present in mammals could possibly even improve these results while still maintaining a high degree of interpretability.

References

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

[2] Daniel Jurafsky and James H Martin. "Speech and language processing (draft)". In: *Chapter 6: Vector Semantics and Embeddings (Draft of December 29, 2021)* (2021).

[3] Yuchen Liang et al. "Can a Fruit Fly Learn Word Embeddings?" In: *arXiv preprint arXiv:2101.06887* (2021).

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

[5] Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. "Linguistic regularities in continuous space word representations". In: *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*. 2013, pp. 746–751.

[6] 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)*. 2014, pp. 1532–1543.

[7] Chaitanya Ryali et al. "Bio-inspired hashing for unsupervised similarity search". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 8295–8306.

[8] Lütfi Kerem Şenel et al. "Semantic structure and interpretability of word embeddings". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.10 (2018), pp. 1769–1779.

[9] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[10] Yuwei Wang et al. "Biological Neuron Coding Inspired Binary Word Embeddings". In: *Cognitive Computation* 11.5 (2019), pp. 676–684.