The hippocampus and language: Word to word prediction in terms of the successor representation

Bachelor's Thesis in Computer Science

submitted
by

Philipp Rost

born 24. May 1996 in Fürth

Written at

Lehrstuhl für Mustererkennung (Informatik 5)
Department Informatik
Friedrich-Alexander-Universität Erlangen-Nürnberg.

Advisor: Paul Stöwer M. Sc., Prof. Dr.-Ing. habil. Andreas Maier, Dr. rer. nat. Patrick Krauß (Neuroscience Lab, University Hospital Erlangen)

Started: 1st June 2022

Finished: 10th July 2022

Erlangen, den 10. Juli 2022

Übersicht Den theoretischen Hintergrund der Masterarbeit bilden die Ort- und Rasterzellen des Hippocampus, die für verschiedenste Aufgaben der Orientierung zuständig sind. Das reicht von abstrakten Zuordnungen wie der Höchstgeschwindigkeit zu einem Fahrzeug auf Grundlage der Motorleistung und des Gewichts bis zur klassischen räumlichen Navigation in einer Stadt oder einem Gebäude. Da diese Resultate bereits per Maschinellem Lernen untersucht wurden, soll diese Arbeit davon handeln, ob diese Methoden auch dazu verwendet werden können, um Sprache zu verarbeiten, damit so ggfl. Rückschlüsse auf die Orts- und Gitterzellen gezogen werden können. Zu diesem Zweck soll die Theorie der Projektiven Karten und deren mathematischer Formulierung der Successor Representation genutzt werden. Um dies zu erreichen, werden mehrere Architekturen eines Neuronalen Netzes untersucht und verschiedene Techniken des Natural Language Processing verwendet, wobei das Hauptaugenmerk auf der Verarbeitung von Büchern liegt, mit denen die Trainingsdaten generiert werden können, da sie plausible Sprachdaten darstellen.

Abstract The theoretical background of my master thesis is founded on the concept of the places and grid cells of the hippocampus, which control different tasks of orientation. They range from classical spatial navigation in a city or building till abstract mappings between velocity and vehicle based on engine power and mass. These results were already examined by the means of Machine Learning. Thus, this work wants to try to extent the methods to process language in hope to gain some conclusions on place and grid cells. For this purpose the Successor Representation as application of the Projective Map theory is implemented by deploying a Neural Network under multiple architectures. Furthermore different techniques of Natural Language Processing are used, because the training data is generated from two books to have plausible language data.

Contents

Chapter 1

Introduction

Understanding and therefore modeling the human brain is a challenge as old as science itself. Through the centuries mankind contemplated and associated the brain as a complex version of the technology they were surrounded by, starting by comparing it with an abacus up to the Human Brain Project (HBP)[1] founded by the EU, which tries to simulate and build a "silicon brain". The first steps toward this venture were taken by Santiago Ramón y Cajal, who was rewarded with the Nobel Prize in 1906 [Noba]. Furthermore, his research includes a full description of a nerve cell and further related concepts, for instance that signals are processed mono-directionally. These results lead directly to Rosenblatt's perceptron [Ros58] and from then on to the nowadays modern field of Deep Learning in Computer Science.

Next to large scaled research projects like the HBP there also exist smaller ones tackling different parts of the brain, for example the hippocampus (Section 2.1 Hippocampus). It plays a fundamental role in forming new memories, processing emotions as part of the limbic system and in positioning, a result awarded with the Nobel Prize in 2014 (Section 2.1 Hippocampus & Section 2.2 Predictive map theory) [Nobb]. This abstract space in which the navigation happens is called cognitive room (Section 2.1 Hippocampus) and Stachenfeld et al. provide an adequate mathematical theory, the Successor Representation (SR) (Section 2.3 Successor Representation), to transfer the concept into a framework to work and do experiments with.

This master thesis tries to extend the application of the SR, which is focused on spatial coordination in [Sta+17] to learning languages. For this purpose some techniques of Natural Language Processing (NLP) are useful, necessary and applied (Section 3.2.1 Data preparation). One motivational hint this plan could work out is given by Stachenfeld et al.,

[1]https://www.humanbrainproject.eu/en/

who showed that their theory doesn't need well structured surroundings like a city but a topological/graphical environment is sufficient to retrieve viable results. Therefore, P. Stöwer did promising first experiments [Stö21], which will be used as foundation to generalize his framework (Chapter 3 Framework) in an attempt to gain more precise results (Chapter 4 Methodology and Results).

Chapter 2

Theoretical Background

2.1 Hippocampus

The hippocampus is located in the brain and part of an old area the archicortex. It has its name from the Greek word for seahorse because it has the shape of one (Figure 2.1). This brain area can be divided into three parts: the dentate gyrus, the cornu ammonis and the subiculum [ORe+20; Gar18].

The hippocampus plays a fundamental role in forming new memories (not preserving them, which is done across the brain) and is highly capable of learning new information fast. Regarding its functions, one was already mentioned: It is the key area when it comes to establish new memories. Patients with a damaged hippocampus lacking this ability will lose spatial and temporal orientation. Furthermore are epilepsy, schizophrenia and Alzheimer's disease connected to this dysfunctional organ [Tre17]. The hippocampus is also important in emotional contexts because it is an integral unit of the limbic system [Gar18]. Another task, and for this thesis the most important one, is navigation/orientation but not just in spatial surroundings but also in an abstract contexts, called cognitive room. Some examples for abstract contexts are: Danger of animals based on their appearance and speed of vehicles based on their weight and engine (Section 2.2 Predictive map theory). To achieve this skill two types of cells in the hippocampus are active: place cells and grid cell. The first one encodes states/positions (one for each cell) and the latter resembles a coordinate system.

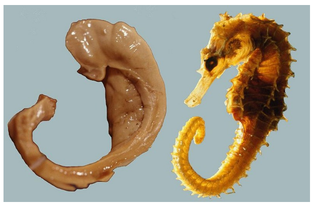
Figure 2.1: Hippocampus and seahorse [Ser10]

Place cells Place cells are irregular distributed across the cognitive room. Their firing is tied to the location of the state, whereby the term location has not always its classic spatial meaning if we navigate in an abstract setting (as mentioned above). The place cell is active in case we encounter the associate state. As seen in Figure **??** different place cells (each is color coded) fire at different positions in the parkour e. g., turquoise is undoubtedly related to the first arch, meaning its activity spikes while the rat passes by. The remark of the thesis lies on place cells.

Grid cells This type of cell can be found in the entorhinal region and satisfies a more general purpose. They are regularly distributed and form a triangular lattice (Figure **??**). It provides raw spatial information in terms of a metric or distance measure the hippocampus integrates with the place cells [ORe+20; Bel+18].

2.2 Predictive map theory

To explain the principle of the predictive map theory introduced in [Sta+17], it is necessary to illustrate the concept of a cognitive room, mentioned before in Section 2.1 Hippocampus. An example is of course a naive navigational task as presented by Stachenfeld et al. and similar to the setting of Figure **??**. The authors even demonstrated that just a topological environment is sufficient to craft a cognitive room and apply the predictive map theory.

The concept becomes far more interesting when talking about experience based cognitive rooms i. e., the speed of vehicles based on weight and engine specifications. This category of a cognitive room also fits the topic of the thesis much better, since it aims to model language not a spatial environment. An illustrating example can be found in Figure **??**. For instance, a “sports car” might be rather lightweight but has plenty of horse power. By using these two characteristics the cognitive room has the shape of a $2d$-plane. For instance while reading about an alien car, it is immediately possible to compare it with different well-known vehicles and draw conclusions about its shape since the cognitive room has enough information to position the car within it. All these decisions of placing new objects in an appropriate context is done by place and grid cells (Section 2.1 Hippocampus). Expanding the example by the firing of cells results in the full illustration given in Figure **??**.

2.3 Successor Representation

According to Stachenfeld et al., our behavior in an open spatial environment, or in general in a cognitive room, e. g., a city, follows the predictive map theory introduced in Section 2.2 Predictive map theory. An active place cell encodes the next/successor state entered by the agent.

To model this or the general setting of predicting future states the SR was developed by a Reinforcement Learning (RL) approach. Furthermore, the SR and the predictive map theory go hand in hand. The latter is an application regarding the former: The authors support the proposition that hippocampal mechanics explained by the predictive map theory, can be described via the SR.

2.3.1 Mathematical Foundation

The basis lies to a broad extent in RL, in formula:

$$V(s) := E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | s_0 = s\right]. \tag{2.1}$$

Here, V resembles a value function, expressed via reward function R, which operates on state s_t, encoded by the sum over t, starting in s. $\gamma \in [0, 1]$ serves as discount factor to control the influence of states reached in distant future. High values permit distal states to play a larger role, whereas smaller values de facto limit the result to neighboring positions[1]. By the reward function is obtained how beneficial the currently visited state s_t is. After the calculation of V the function can be decomposed into a more intuitive representation consisting of a state matrix M, called the SR-matrix, and the known reward function R:

$$V(s) = \sum_{s'} M(s, s') \cdot R(s'). \tag{2.2}$$

The first argument of M specifies the row, the latter the column. Each cell contains the discounted expected number of times the agent visits state s' starting from s. Additionally, Stachenfeld et al. mention that the SR-matrix can be derived from a transition probability

[1]Short mathematical explanation: $p^t \xrightarrow{t\to\infty} 0$ for $p \in [0, 1)$, the greater p the slower happens the approach of the limit. For $p = 1$ the sequence is constant. In our case every state is taken into account equally.

matrix T for the positions s [Sta+17]. Having T, it follows

$$M = \sum_{t=0}^{\infty} \gamma^t T^t, \tag{2.3}$$

which is geometric series and converges for $\gamma < 1$ towards

$$(I_n - \gamma T)^{-1}, \tag{2.4}$$

where I_n is the corresponding identity matrix.

Although defining all formulae by infinite sums, it is seamlessly possible to calculate the SR-matrix in Equation (2.3) up to a finite index or starting at an arbitrary t i. e., $t = 1$. Doing so makes sense in a language environment. The identity matrix would imply that a word can follow itself, which is extremely rare[2]. Therefore, the first summand will always be γT, where T is calculated by a Neural Network (Section 3 Framework). If the indices in Equation (2.3) are altered, the limes of the geometric series no longer applies directly and has to be adjusted by subtracting the first summands from Equation (2.4). Since the SR and thus the depicted formulas, especially the SR-matrix and the transition probability matrix, are policy dependent which is reflected by the training data.

By definition, matrix M reveals all successor states with their particular probability because the summation combines the following positions, which are calculated by exponentiation, into one matrix. By examining a row (Figure **??**) e. g., row k, it is possible to follow all paths starting from state k.

One advantage of the SR, i. e., describing the model by the SR-matrix M, is its high flexibility regarding the evaluation of different reward functions given by the Equation (2.2). The value of a state s can be calculated in an instant with a different reward function and no relearning is necessary.

SR and grid cells Although not further discussed in the thesis but an interesting claim of Stachenfeld et al. is that the eigenvalue decomposition of the SR-matrix reveals the grid cell structure. They provide Supplementary information depicting many examples [Sta+17].

[2]Although sentences like “Ich hoffe, dass das das Richtige ist.” do occure in german.

2.3.2 Example for the Successor Representation

This subsection is dedicated to fill the concept of the SR and the SR-matrix M with some intuition. Stachenfeld et al. simulated a linear spatial environment build by six states with a simple policy merely consisting of two actions the agent can apply: Going one step to the right or pausing.

Rows In this scenario a plot of single rows of M can be found in Figure **??**. By the upper half of Figure **??**, examining the row of s^1, it is possible to deduce that the agent will most probably pause in its current position with vanishing values for distal locations. A different point of view results for the part of $M(s^5, s^i)$. It is obvious that going backwards is nothing to reckon with since the numbers for transitioning distribute over s^5 and `goal` by slightly favoring the former. This behavior was expected by the policy.

Columns It is also worth analyzing the column of M, called *place field* by Stachenfeld et al. (Figure **??**). Having the policy in mind it is no surprise that the values $M(s^i, s^5)$ ascend in parallel to the index i. The probability for entering s^5 grows by approaching it. In addition the plot shows how s^5 is probably reached best, simply by passing via s^3 and s^4. This might seem obvious but in a more complex cognitive room the graph won't look as ordinary and therefore will contain more further distributed information.

2.4 Multidimensional Scaling

The goal of Multidimensional Scaling (MDS) is to calculate a m-dimensional mapping, $m < n$, of a given point cloud in $\mathbb{R}^n$ that preserves the original distances as good as possible [Has+17]. The result of the calculation is unique modulo rotation and scaling. Therefore, it is based on a metric not exact coordinates. MDS is used to analyze similarities between the rows of the SR-matrix by determining clusters in the graph.
The algorithm is simple and only uses basic linear algebra. MDS works with a distance matrix D where each entry is equal to d_{ij}^2, the squared distance between two points $x_i,\ x_j \in \mathbb{R}^n$, whose coordinates are (in principle) unknown. By double centering D it is possible to calculate the matrix product $X^\top X$, where X bears the coordinates in the desired dimension [Rie20]. Double centering means multiplying by a matrix $C := I_n - \frac{1}{n}J_n$,

where J_n is a $n \times n$-matrix of ones:

$$\underbrace{-\frac{1}{2}CDC}_{B:=} = X^\top X, \tag{2.5}$$

The centering matrix C has, after a multiplication with a column vector, the same effect of subtracting the mean of all components from the vector itself.
In the next step, the m largest eigenvalues B are calculated along with their corresponding eigenvectors. Finally, the m-dimensional coordinates are determined:

$$X_m = E_m V_m^{1/2}, \tag{2.6}$$

where E_m contains the m eigenvectors and V_m is a m-dimensional diagonal matrix with the associated eigenvalues.

2.5 Metric for quantifying the results

Throughout the presentation of the results in Chapter 4 Methodology and Results many matrix and MDS plots are used. Sometimes they give a clarifying visual response but not in all cases. When comparing different approaches, images lack the needed objectivity and plausible criteria to rate the outcomes. To tackle this flaw, a metric was developed to have the possibility to draw objective conclusions.

Since Neural Networks on languages are trained, a measure on the grade of the closeness to the real counterpart is necessary, in the following referred by "ground truth (distribution)". The mathematical objects are in both cases squared matrices of dimension $n \in \mathbb{N}$ built by transposed probability vectors. Nevertheless, the presented mapping is made for $n \times m$-matrices. Depending on the model type, the ground truth vectors are 1-hot-encoded vectors or share different fractions across all entries (Section 3.2 Word to word models).

Hence, the starting positions for the metric are probability vectors. The obvious way to quantify the results is by taking the euclidean norm d of the difference of the ground truth and the prediction. Consequentially, d takes values between 0 and $\sqrt{2 \cdot n}$ because the maximal difference for each row is $\sqrt{2}$ and there are n in total. $\sqrt{2}$ is derived by the following nonlinear program

$$\begin{aligned} \max \quad & \|\boldsymbol{x} - \boldsymbol{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \\ \text{s.t.} \quad & \sum_{i=1}^{n} x_i = 1, \ \sum_{i=1}^{n} y_i = 1 \\ & x_i, y_i \in [0, 1], \end{aligned} \tag{2.7}$$

which is solved by 1-hot-encoded vectors for $\boldsymbol{x}$ and $\boldsymbol{y}$ where $x_i = 1 \neq y_i$ for a $i \in \{1,\ ...,\ n\}$. Or to put it bluntly, the difference takes its highest values for all scenarios in which the $\boldsymbol{x}$ and $\boldsymbol{y}$ are perpendicular and have a maximal euclidean norm, which means being a 1-hot-encoded vector. This relation is present n-times for the ground truth matrix and the learned one implying that the maximal difference is $\sqrt{2 \cdot n}$.

Finally, it is possible to define the metric on the set $\mathcal{P}$ of $n \times m$-probability matrices:

$$d_A \colon \mathcal{P} \to [0, 1], \qquad L \mapsto \frac{1}{\sqrt{2 \cdot n}} \|A - L\|_2, \tag{2.8}$$

where A describes a fixed matrix in $\mathcal{P}$. In the scope of this work the ground truth will play the role of A. By d_A the learned matrix L is mapped to 0 if it matches the ground truth distribution perfectly and to 1 if the rows satisfy the conditions mentioned above.

Chapter 3

Framework

After setting up the theoretical basis for the tools needed to start the experiments, the code framework will be introduced. In general it is a supervised dense Neural Network written in `python` [Van+09] with the help of `keras` [Cho+15] and `numpy` [Har+20]. The process is divided into two phases: One training phase and one for result visualization. The goal was to be able to calculate a decent SR-matrix showing visible clusters in the MDS-plot. To retrieve the SR, a Neural Network is trained, whose predictions serve as transition probability matrix T. If not otherwise stated a discount factor $\gamma = 0.5$ is used.

Different scenarios were tested, hence various models were configured having distinct features e. g., some work with 1-hot-encoded vectors other use word vectors or made up rules and datasets.

3.1 First Model and Architecture

The first class of networks augments the results in [Stö21]. P. Stöwer's models rely on predefined rules like

- `Adjective → Noun`
- `Verb → Adjective`
- `Personal Pronoun → Verb`
- `Question word → Personal Pronoun`

for building the dataset backed by a word database containing the corresponding information. Starting point is the cognitive room, which consists of a list reflecting the whole data. The training data was crafted in accordance by randomly choosing respectively one of the four rules above and within the word class an example. This information is used to initialize a 1-hot-encoded vector and is done for input and output of the network.

The goal was to attain results on the behavior of the model if it is extended by more rules and words. One can imagine this type of model as a graph (Figure **??**).

3.2 Word to word models

The harder challenge lies in unannotated texts without a paradigm for pairing words of an example data set. In this manner a language normally occurs and is learned by humans. A priori one has to expect results of poorer quality in comparison to the configuration of Section 3.1 First Model and Architecture because they were tailored and NLP comes always with uncertainties.

3.2.1 Data preparation

The data is extracted from two books, namely “Gut gegen Nordwind” written by Daniel Glattauer in german [Gla06] and from Jostein Gaarder “Sophie’s World” [Gaa96] in english. Two languages were chosen since German comes in general with a high degree of freedom regarding word order, whereas english is more restrictive. This distinction may be important, since analyzing successive words is fundamental for this work. Because the books are available as `pdf`-file, the python module `pymupdf` [McK$^+$] is used to generate a simple `String` containing the whole text, which is afterwards parsed by `spacy` [Hon$^+$17]. This is a powerful tool in the area of NLP and some techniques are indispensable for further analysis, mainly

- Tokenization: segmenting text into words, punctuations marks etc.
- Part-of-speech (POS)-Tagging[1]: assigning word types to tokens, like verb or noun
- Lemmatization: assigning the base forms of words[2]
- word2vec [Mik$^+$13a; Mik$^+$13b]: calculating a vector representation with real values of a word, in the following called *word vector*.

[1]More information on [dMar$^+$]

[2]For example, the lemma of “was” is “be”, and the lemma of “rats” is “rat”.

Additionally, a mechanism was implemented to extract an exact number of words to have a equal sized foundations in both languages. The training data consists of word pairs in their occurring order, for instance the sentence

> Goethe remarked about Alexander von Humboldt to friends that he had never met anyone so versatile.[3]

gets tokenized, lemmatized and coupled having

```
("Goethe", "remark"), ("remark", "about"), ("about", "Alexander"), ...
```

where the first component serves as input and the second one as supervised output.

Clearly, not the actual word is fed into the Neural Network but numerical representations: either a 1-hot-encoded vector or a word vector. To construct the former the concept of the cognitive room is applied by building a list containing all words of the text i. e., one word resembles one state and states are encoded by place cells. To learn the transition probability matrix, as proclaimed at the beginning of the chapter, one has to transform the prediction of the Neural Network into a probability vector via division by its sum. This processing is done not during training because it is supervised via 1-hot-encoded vectors.

3.2.2 1-hot-encoded vector approach

This configuration follows the principles of the first model (Section 3.1 First Model and Architecture) by using 1-hot-encoded vectors as input and output but there are no invented grammatical rules anymore. The training data is now directly related to concrete words and not to a word class. An illustration of the data structure is given in Figure **??**.

3.2.3 Word vector approach

The Neural Network takes word vectors, a $300d$-vector of real numbers, as input and omits them during training. Word vectors are calculated by `spacy`. To be precise, this step has two stages. Building the training data is easy because it is effortlessly possible to retrieve the real valued vector given a word. Since predictions aren't (and can't be) as accurate as the results `spacy` computes, it is impossible to query a dictionary or database to reshape the exact word. For such situations, the module offers the option to retrieve a list with the $n \in \mathbb{N}$ closest words. This list includes the n words whose vector representations have the smallest euclidean distance to the desired vector i. e., the prediction.

[3]Sentence taken from [Wul16]

Table 3.1: List of relevant POS-tags including examples and short definition.

POS-tag	Definition & Example	POS-tag	Definition & Example
`ADJ`	Adjective: educated, hot	`VERB`	Verb: to run, to drink
`ADV`	Adverb: easily, everywhere	`DET`	Determiner: this, a, no
`NOUN`	Noun: car, bottle	`PART`	Particle: 's, not
`AUX`	Auxiliary: to have, should	`ADP`	Adposition (Pre- & Postpositions): in, on
`PRON`	Pronoun: she, ours	`REST`	Rest (Container for Conjunctions and additional residuals): and, if

In the next step, a check is performed whether the word is part of the book i. e., the cognitive room. If so, the euclidean distance is taken as entry in the Transition Probability Matrix T which will undergo a row-wise transformation to fit the criteria of a transition probability matrix. One disadvantage is an ambiguous prediction and therefore a less sharp T. But in comparison to the 1-hot-encoded vector a word vector bears a lot more information which hopefully can be exploited by the neural network. The data structure is shown in Figure **??** next to the 1-hot-encoded vector equivalent.

3.3 Average approach

Because the models become enormous and especially thus the evaluation difficult, this configuration aims to analyze the results on a rougher scale by taking averages of the predictions. While collecting the training data, the POS-tag of each one is saved and after training the cumulative outputs of a word class prediction are taken i. e., all 1-hot-encoded vectors of a `VERB` are mapped by the Neural Network, averaged into one vector and then checked for the most probable POS-tags (Section 4.3 Averaging models).

POS-tags were mentioned before in Section 3.2.1 Data preparation and are just another term for word classes. In detail, a subset of the UniversalDependencies POS-tags [dMar$^+$] are used (Table 3.1) because this project provides a rich dataset, good documentation and its guidelines are implemented by `spacy`.

Chapter 4

Methodology and Results

4.1 First Model and Architecture

Starting with the predefined structures as mentioned in Section 3.1 First Model and Architecture i. e., a rule set was given which is equivalent to graphical model, where the rules describe the edges. The training data was composed by randomly choosing one of the five rules and the input as well as output word. The Neural Network was quite shallow because it contained only one hidden layer. By using eight words from each class the rules are easily learned (Figure **??**). Since the rows encode the states, it is expected in the terms of the successor representation that the next position shows a high activation. Checking this behavior is simple because the environment (Section 3.1 First Model and Architecture) is clearly defined. The `Noun` part of Figure **??** is smeary because there is no rule for consecutive state (in graph theory it corresponds to a sink). The undefined behavior is also revealed in the MDS illustration where all nouns are distributed between the other word classes. By calculating the SR for additional time steps, the result incorporates all following rules or states (Figure **??**). These matrices are interesting for analyzing the value function V (Equation (2.2)) which is beyond the scope of this work.

The principle also works out when using more rules and a larger data set. The enhancement is done by introducing four new rules

- `Question word → Verb`
- `Noun → Personal Pronoun`
- `Adverb → Possessive Pronoun`

- `Personal Pronoun → Adverb`,

more words for the already existing categories and two new word classes (`Adverb` and `Possessive Pronoun`). The result in Figure ?? is in clarity similar to Figure ?? but without having an obvious blurry row even though the word class `Possessive Pronoun` is without successor as `Noun` in the trial before but some similarities to `Question word` are detected. The only characteristic they seem to share is having exactly one edge. The clusters are in the MDS apparent. Although artificial, conceptually they serve as a standard to reach for the upcoming configurations.

4.2 Word to word models

The harder challenge lies in processing natural languages. For this purpose two different approaches, namely 1-hot-encoded vector approach and Word vector approach were examined. The ambitious goal was to learn or get a sense of the grammatical structure of the text. In practice this means that after feeding e. g., an adjective into the Neural Network it should propose words or word classes which follow this word. In the following paragraphs an schematic outline of the model will be given i. e., few data was involved to achieve more reasonable diagrams. The full book covers multiple thousands of states, leading to a sparse squared matrix in this dimension. Thus, depicting matrices no longer makes sense.

To be able to evaluate the results a ground truth distribution was calculated for gaining a qualitative and quantitative measure. In Figure ?? is one depicted. The data set is labeled on the axis. There is also a MDS plot next to it because on larger scaled models (> 500 words) it is impossible to get visual feedback just by looking at the matrix plot. The hope is that clusters of the same color emerge, as seen in Section 4.1 First Model and Architecture, because some meta word pairs, for example `Noun → Verb` i. e., `Alice → goes` and `wood → breaks`, should share some features and be mapped close to each other. Furthermore, similarities are to some extent also visible by looking at the MDS.

Having a reference now, the model was set up for training. In Figure ?? is an emblematic transition matrix illustrated. Whereas in Section 4.1 First Model and Architecture it was easy to recognize the rules in the SR, this will no longer be possible due to the sheer amount of states. Hence, the cluster plot is more important to gain visual intuition for the results.

Table 4.1: Trained models with metric values regarding their corresponding ground truth. “german” and “english” refer to the book which provided the data set (s. Section 3.2.1 Data preparation). The associated MDS plots for a qualitative feedback can be found in Figure **??** and Figure **??** of Appendix B Cluster plots of word to word models.

Version	Metric
german, 1-hot-encoded vector	0.08
german, word vector	0.74
english, 1-hot-encoded vector	0.10
english, word vector	0.78

4.2.1 Word to word models: Evaluating the results

To achieve the best result four configurations were tested and compared in Table 4.1 by the metric introduced in Section 2.5 Metric for quantifying the results. Surprisingly, the german and english 1-hot-encoded vector versions nearly don’t differ. Not only regarding the model used to collect the data illustrated in this thesis but in general. Due to the beforehand mentioned freer word order german incorporates, a better score for the english based models was expected.

It is obvious that the models using word vectors perform quite bad compared to the 1-hot-encoded vector variants. The cause may be on the one hand that language is a structural complex field and on the other that too many uncertainties are attached to word vectors, learning is more difficult since 300 distinct values are involved, whereas a 1-hot-encoded vector may be quite accurate if the learned version has its maximum near or at the index where the input vector is equal to 1. Also back-calculating the output to words is a process of compromises because it is guaranteed that `spacy` doesn’t provide a word vector with the exact same components and it is necessary to limit on the n closest word vectors/words. The disappointing outcome of the word vector model is furthermore frustrating since it is more probable that the hippocampus doesn’t process signals which are close to 1-hot-encoded vectors (or an analogy of it) but rather multiple stimuli decoding different characteristics which is thus more related to a word vector.

Additional configurations While searching for the best parameters, not just the four versions mentioned were examined. Most of the time was consumed by finding a promising setup. This is reflected by some parameters the framework provides (e. g., `nmb_hidden_layers`, Section D Training parameters) which aren’t necessary to reproduce the results presented

Table 4.2: Schematic sentences of the book used in Figure ??.

Rule	Sentence
`ADP → NOUN`	Sie hat mich `zum Schmunzeln` gebracht.
`ADP → PRON`	Es war überaus angenehm, sich `mit Ihnen` zu unterhalten.
`ADP → DET`	Ich bin nämlich eine gebürtige Linkshänderin, die `in der` Schule, [...]
`ADJ → NOUN`	Lieber Leo, ich habe drei `fürchterliche Tage` hinter mir.

in this work. A subset with illustrations and short documentation of all variants can be found in Appendix A Additional configurations.

4.3 Averaging models

Because large scale word to word models seem to lack characteristics for a proper evaluation and interpretation of the results, the average approach was developed (Section 3.3 Average approach). The main idea is to rearrange the output of the formerly presented models, because general patterns like `Noun → Verb` or `Determiner → Adjective` appear significantly more frequently than certain instances as `sun → shines` or `an → old`. By averaging the predictions one might be able to catch the structure in general.

Since Section 3.3 Average approach only provided a rough description of the principle, it shall be extended in the following. After the collective prediction of one word class, all outputs are averaged having one vector resembling the complete data in the dimension of the cognitive room showing the associated frequencies. In the next step the n indices (in practice 10 was used) with the highest values are checked for their POS-tag. So, if index i is one of the n highest and encodes the word `fish`, its POS-tag i. e., `NOUN` will be counted. Finally, one has constructed a vector for all word classes where each component resembles the probabilities for the successor word class. Some final numbers are illustrated in Figure ??.

To get an idea of the learned results, some valid sentences are provided in german (the language of the training data of Figure ??) in Table 4.2. This short sanity check implies that there is no cacophony learned and the outputs are reasonable.

4.3.1 Averaging models: Evaluating the results

The same configurations as in Section 4.2.1 Word to word models: Evaluating the results were processed. By “configurations” is subsumed: german and english, 1-hot-encoded vector

Table 4.3: Averaged means of the difference between prediction and ground truth. As expected, the word vector versions have higher scores than the 1-hot-encoded vector counterpart. These values can't be used for a comparison with Table 4.1 because the underlying metrics are different.

Version	Mean in 10^{-2}	Standard deviation in 10^{-2}
german, 1-hot-encoded vector	7.3	2.0
german, word vector	14.0	2.1
english, 1-hot-encoded vector	8.1	3.3
english, word vector	10.2	3.6

or word vector version and the same number of epochs and words used for training. After the unsatisfactory performance of the word vector approaches (Table 4.1) everything else than a similar outcome would be surprising. A detailed illustration of the results offers Figure **??**. By this plot it is also possible to talk about the outcomes a bit more specifically. All means are to some extent equal but the standard deviations differ. The free word order of german doesn't seem to be a problem because both models learn better than their english counterparts (Table 4.3). Surprisingly, there are some difficulties with adjectives (except for Ger. OHE), although adjectives are tied to `NOUN` and `ADJ` (Figure **??**).

Sadly, the prepared visualizations i. e., Figure **??** and Table 4.3, lack the ability to grasp the full picture: The word vector configurations seem to work by checking the means for each POS-tag, although using german scratches the edge of the phrase, but when inspecting the matrices as a whole as in Figure **??**, it is apparent that they don't. Whereas using the english book the model degenerates to the identical distribution and the german one contemplates `NOUN` as its personal 42.

Chapter 5

Conclusion

Since no data from valid neural scans researching the same or an similar topic was provided, the project is heavily theory based (in comparison to [Sta+17]) which makes the interpretation of the results a priori not easy because a quantitative and qualitative frame of reference is missing. There were sharp results produced in Section 4.1 First Model and Architecture but they are too artificial to draw relevant conclusions regarding the objective and neuroscientist won't collect data as clear.

While trying to reproduce them i. e., getting as close as possible, many architectures were unsuccessfully tested and evaluated as mentioned in Appendix A Additional configurations. Therefore, the process of finding a proper one consumed many weeks with discussions between my advisor and me. Maybe, the goals were slightly too ambitious. Additionally, they more or less led to the same results covered in this thesis, especially in Section 4.2 Word to word models. Although, the values calculated in Table 4.1 are relatively low and close to 0, which means a perfect fit according to Section 2.5 Metric for quantifying the results, the metric d_A itself isn't justified for more than internal comparisons of the configurations. It was developed to have a sensible measure on the results because plots of high dimensional sparse matrices, which are just monochromatic squares, don't convince. From the figures in Table 4.1 & Table 4.3 can be deduced that models training with word vectors have an unsatisfactory performance in comparison to their equivalents working with 1-hot-encoded vectors. This is unsatisfying for two reasons: Firstly, getting the approach working costed much time because the implementation of the mechanics `spacy` offers and integrating them into the concept of the cognitive room were the most elaborate part in the process of building the framework, and secondly because the input the hippocampus receives is probably closer to a word vector than to a 1-hot-encoded vector. Translating it into a neuroscience behavior, it can be compared with receiving plenty of signals as input against processing one (strong) activation from another cell.

By presenting the topic in front of my colloquium, further approaches were gathered. One of them, the idea of averaging the predictions of one word class and analyzing the outcome, paid off (s. Section 3.3 Average approach & Section 4.3 Averaging models). The results found there can function as addition to the plain word models because they prove that these models can grasp grammatical structure. Therefore, they can provide visual feedback even in large dimensional contexts and by calculating the ground truth distribution there is a useful reference.

But nevertheless, the 1-hot-encoded vector variant can serve as foundation for further (minor) research, for instance developing a better metric to have a mathematical notion of encoding a good and objective value or tweaking the learning with word vectors due to the better compatibility with hippocampal functions. Of great value would be collected data from an analogue survey conducted by neuroscientist i. e., analyzing the activity of the hippocampus, the place and grid cells while participants process new pieces of language. Then is a viable environment as in [Sta+17] is given and the theory may be expanded to language related topics as it is to spatial navigation.

Appendix A

Additional configurations

To draw a full picture, plenty of approaches which had the goal to improve the results will be mentioned in this chapter. Sadly, no one changed the outcome by any means. Facing this presented an enormous obstacle while researching. Some of them will be presented shortly in this chapter. In all cases it is obvious that these configurations were dead ends.

A.1 Multiple hidden layers

Different numbers of layers ranging from 1 to 100 were tested, some example results will be depicted.

A.2 Many epochs and multiple hidden layers

The example outputs stem from a model which trained with sixfold epochs and 40 hidden layers.

A.3 Using word vectors to learn an 1-hot-encoded vector

This configuration is combination of two mainly used in the thesis. It uses word vectors as input and 1-hot-encoded vectors as output i. e., it uses heterogeneous structured training data.

A.4 Multiplying the training data

The goal of multiplying the training data i. e., concatenating the training data n times with itself, was to have the opportunity to see the training data more often during one epoch.

A.5 Calculating high time steps

One idea was to calculate high time steps of the SR hoping the irregularities even out in distant future.

A.6 Predict only the most frequent words

Similar to Section A.4 Multiplying the training data, the most frequent words of the text are seen more often by the network. Hence it might be able to learn these inputs better than ordinary ones.

Appendix B

Cluster plots of word to word models

Visualizing matrices training with more than 500 words doesn't make sense, because the result is high dimensional and sparse. To get at least some visual feedback, MDS plots are calculated. To avoid a crowded picture, a compromise had to be made: depicting solely the POS-tag means a manageable overview but information on single words gets lost.

Appendix C

Barplots of the average approach

The barplots illustrated here stem from a 1-hot-encoded vector model using a german book to collect the training data. It achieved the best value ($7.3 \cdot 10^{-2}$) of all models tested. The scores are listed in Table 4.3. If a POS-tag doesn't appear in the plots i. e., having no green or blue bar, this means that it doesn't succeed the depicted POS-tag. The explanation of the POS-tags can be found in Table 3.1.

Appendix D

Training parameters

In case my results shall be reproduced with the framework I programmed, the parameters of Table D.1 were used. For the results in Appendix A Additional configurations the numbers differ e. g., more `epochs`, a higher `nmb_hidden_layers` or `nmb_concatenations`. The exact factor is mentioned there.

Table D.1: Training parameters of the network.

Parameter	Value
Learning rate `lr`	0.1
`epochs`	4000
`batch_size`	100
`pages`	200
`nmb_tokens`	1500
`nmb_hidden_layers`	1
`nmb_concatenations`	1
`book_name`	0 (german), 1 (english)

List of Abbreviations

HBP	Human Brain Project
MDS	Multidimensional Scaling
NLP	Natural Language Processing
POS	Part-of-speech
RL	Reinforcement Learning
SR	Successor Representation

List of Figures

List of Tables

Bibliography

[Bel+18] J. L. S. Bellmund, P. Gärdenfors, E. I. Moser, and C. F. Doeller. Navigating cognition: spatial codes for human thinking. *Science*, 362, 6415, November 2018. DOI: 10.1126/science.aat6766. URL: https://science.sciencemag.org/content/362/6415/eaat6766 (cited on p. 4).

[Cho+15] F. Chollet et al. Keras. https://keras.io, 2015 (cited on p. 11).

[dMar+] M.-C. de Marneffe, C. Manning, J. Nivre, and D. Zeman. Universal pos tags. URL: https://universaldependencies.org/u/pos/index.html. Online, accessed on June 19th 2022 (cited on pp. 12, 14).

[Gaa96] J. Gaarder. *Sophie's World. A novel about the history of philosophy.* Berkley, New York, 1996. ISBN: 0425152251 (cited on p. 12).

[Gar18] N. Garzorz-Stark. *Basics Neuroanatomie.* Urban and Fischer/Elsevier, 2nd edition, 2018. ISBN: 9783437424588 (cited on p. 3).

[Gla06] D. Glattauer. *Gut gegen Nordwind.* Deuticke, Vienna, 2006. ISBN: 9783552060418 (cited on p. 12).

[Har+20] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2 (cited on p. 11).

[Has+17] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, Heidelberg, 2017 (cited on p. 7).

[Hon+17] M. Honnibal and I. Montani. spaCy 2: natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. 2017. URL: https://spacy.io/ (cited on p. 12).

[McK+] J. X. McKie and R. Liu. Pymupdf: pymupdf (current version 1.19.6) is a python binding with support for mupdf (current version 1.19.*), a lightweight pdf, xps, and e-book viewer, renderer, and toolkit, which is maintained and developed by artifex software, inc. URL: https://github.com/pymupdf/PyMuPDF. Online, accessed on April 10th 2022 (cited on p. 12).

[Mik+13a] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. October 2013. URL: https://arxiv.org/abs/1310.4546. Online, accessed on June 7th 2022 (cited on p. 12).

[Mik+13b] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. January 2013. URL: https://arxiv.org/abs/1301.3781. Online, accessed on June 7th 2022 (cited on p. 12).

[Noba] NobelPrize.org. The nobel prize in physiology or medicine 1906. URL: https://www.nobelprize.org/prizes/medicine/1906/summary/. Online, accessed on June 5th 2022 (cited on p. 1).

[Nobb] NobelPrize.org. The nobel prize in physiology or medicine 1906. URL: https://www.nobelprize.org/prizes/medicine/2014/summary/. Online, accessed on June 7th 2022 (cited on p. 1).

[ORe+20] R. C. O'Reilly, Y. Munakata, M. J. Frank, and T. E. Hazy. *Computational Cognitive Neuroscience*. Open Textbook, freely available, 2020. URL: https://compcogneuro.org/. Site of the lab: https://ccnlab.org/ (cited on pp. 3, 4).

[Rie20] C. Riess. Pattern analysis, lecture 02i: multi-dimensional scaling, Faculty of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg in Erlangen, 2020 (cited on p. 7).

[Ros58] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65, 1958. DOI: https://doi.org/10.1037/h0042519 (cited on p. 1).

[Ser10] L. Seress. Hippocampus and seahorse. 2010. URL: https://commons.wikimedia.org/wiki/File:Hippocampus_and_seahorse_cropped.JPG. Online, accessed on May 27th 2022, License: https://creativecommons.org/licenses/by-sa/3.0/ (cited on p. 3).

[Sta+17] K. L. Stachenfeld, M. M. Botvinick, and S. J. Gershman. The hippocampus as a predictive map. *Nature Neuroscience*, November 2017. DOI: 10.1038/nn.4650. URL: https://www.nature.com/articles/nn.4650 (cited on pp. 1, 4, 6, 21, 22).

[Stö21] P. Stöwer. The hippocampus and the successor representation – an analysis of the properties of the successor representation, place- and grid cells, Faculty of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg in Erlangen, April 2021 (cited on pp. 2, 11).

[Tre17] M. Trepel. *Neuroanatomie*. Elsevier, München, 2017. ISBN: 9783437412882 (cited on p. 3).

[Van+09] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN: 1441412697 (cited on p. 11).

[Wul16] A. Wulf. *The Invention of Nature. The Adventures of Alexander von Humboldt*. The Lost Hero of Science. John Murray, London, 2016. ISBN: 9781848549005. URL: https://www.andreawulf.com/ (cited on p. 13).