RETENTION BASED AUTOREGRESSIVE MODELS FOR MODELLING NEURAL DYNAMICS

BY

ABHINAV MURALEEDHARAN

A thesis submitted in conformity with
the requirements for the degree of
Masters in Engineering
Graduate Department of Institute for Aerospace Studies
University of Toronto

ABSTRACT

Retention based autoregressive models for modelling neural dynamics

Abhinav Muraleedharan

Masters in Engineering

Graduate Department of Institute for Aerospace Studies

University of Toronto

2023

In this work, we present Retention, a novel autoregressive model for generative modelling of sequences. Unlike Transformer based autoregressive models, retention scales linearly with respect to context size. We apply retention based models for modelling neural dynamics and achieve SOTA performance in neural modelling and behaviour decoding.

To Mom,

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my thesis supervisor, Prof. Prasanth Nair, for his invaluable guidance, unwavering support, and endless knowledge throughout the completion of my master's thesis. Prof. Nair's insightful feedback and constructive criticism played a pivotal role in shaping the direction of my research. I thoroughly enjoyed our discussions on various ideas, and his mentorship greatly enriched my academic experience.

I would also like to extend my appreciation to my co-supervisor, Taufik Valiante, for his motivating presence and the engaging discussions we had during the course of this research. Mr. Valiante's expertise and enthusiasm for the subject matter inspired me to tackle challenges with a fresh perspective.

Furthermore, I am grateful to both Prof. Nair and Mr. Valiante for not only guiding me in my academic pursuits but also providing valuable career advice that will undoubtedly influence my future endeavors.

This acknowledgment would be incomplete without expressing my deepest gratitude to my parents. Their unwavering support, encouragement, and belief in my abilities have been the cornerstone of my academic journey. Their sacrifices and love have fueled my determination, and I am profoundly thankful for their presence in my life.

Thank you all for your invaluable contributions and support.

PUBLICATIONS

This work has not been published yet.

CONTENTS

1 INTRODUCTION

Hard problems inspire the creation of novel algorithms. These novel algorithms then find application in various contexts, distant from the original application which it was designed for. Understanding the human brain, specifically its dynamics and the relationship between dynamics of the brain and behaviour is a hard problem. Unraveling the dynamics of the brain holds the key to understanding the neural mechanisms underlying these processes. Beyond modeling neural activity, elucidating how such activity correlates with an organism's behavior is crucial for developing Brain-Computer Interfaces, clinical treatments for conditions like epilepsy, depression, and other neurodegenerative diseases. In this thesis, we develop scalable methods for modelling neural dynamics and relationship between the dynamics of the brain and behaviour. Although methods in this thesis are developed specifically for neural data, we believe that our approach would find application in diverse sequence modelling tasks in language, finance and engineering.

Machine learning techniques have played a pivotal role in modeling brain dynamics, and modeling the correlation between neural dynamics and behavior of an animal. In [POC+18], Pandarinath et al. introduced LFADS, an RNN-based method to infer latent dynamics from neural data. More recently, transformer-based models [VSP+17, GZ22], have been applied to learn neural dynamics and behaviour model. In [YP21], Pandarinath et al applied transformer-based models to learn neural dynamics without an explicit dynamical model. While LFADS and NDT (Neural Data Transformers) were focused on learning neural dynamics from single trial recordings, Azabou et.al recently introduced POYO [AAG+23], a transformer-based model to learn neural dynamics from multi-session neural recordings.

Although transformer-based models have shown remarkable success in general language modelling tasks and more recently in learning population dynamics of neurons, they exhibit poor scaling properties especially when applied to neural spiking data. Furthermore, unlike text data, neural recording probes sample on the order of kHz, and hence are characterized by high temporal resolution. This unique temporal aspect of neural

spiking data presents a challenge for transformers, which are originally designed for sequential data but may struggle with the high-frequency nature of neural signals. The transformer's poor scaling properties become particularly evident when recording from a large number of neurons simultaneously, as the number of potential firing patterns exponentially increases with the number of neurons.

In this work, we introduce a new class of autoregressive models to overcome limitations imposed by the architecture of attention-based transformer models. Our model has an unbounded context length and hence can capture long-range dependencies in the time series dataset. Furthermore, the complexity of training and inference of the parametrized model is independent of the context length, and hence our approach is computationally more efficient when compared to transformer-based autoregressive models.

2

PROBLEM STATEMENT

Imagine we are recording data from D neurons distributed across different regions of the brain. Let $x(t_i) \in \mathbb{R}^D$ denote the observed neural activity at timestep t_i and let y_i denote the observed behaviour of the animal at timestep t_i. From the time series dataset $\mathcal{D} = \{(x_i, y_i, t_i)\}_{i=1}^N$ of neural recordings, our goal is to construct:

- A predictive model of underlying brain dynamics
- A probabilistic model to predict behaviour of the organism at time $t+1$ given brain recordings until timestep t.

More formally, let's assume that the spiking activity is generated by an underlying non-stationary stochastic process defined by $p_t(x)$.

$$x(t) \sim p_t(x) \tag{2.1}$$

The probability of observing a sequence of neural recordings and behavior can be expressed as:

$$p(\{x_1, y_1\}, \{x_2, y_2\}, \{x_3, y_3\}, ..) = \lim_{N \to \infty} \prod_{i=1}^{N} p(\{x_i, y_i\} | \{x_1, y_1\}, \{x_2, y_2\}, ..\{x_{i-1}, y_{i-1}\}) \tag{2.2}$$

In the context of neural recordings, it is convenient to assume that the neural recording data and behavior can be modelled with separate probability distributions of the form:

$$\prod_{i=1}^{N} p_d(\{x_i\} | \{x_1\}, \{x_2\}, ..\{x_{i-1}\}) \tag{2.3}$$

$$\prod_{i=1}^{N} p_b(\{y_i\} | \{x_1\}, \{x_2\}, ..\{x_{i-1}\}) \tag{2.4}$$

Specifically, we assume that that the neural observed neural spiking data at timestep t_i is not dependent on the behavior variables in the preceding timesteps. Probability distributions of this nature have been extensively investigated in the field of language modeling. In conventional autoregressive frameworks, the approximation of conditional distributions often involves the utilization of parameterized models constrained by a finite

context limit [VSP+17]. While autoregressive models of this kind have been extremely successful in generating plausible language [RNS+18], they still struggle to capture long-range dependencies due to the finite context length limit[Hah20]. Furthermore, the complexity of training and inference of transformer-based models is $\mathcal{O}(N^2)$, where N is the context length of the transformer model.

3

METHODS

In this chapter, we describe the theory behind Retention based autoregressive models and methods for training retention based models.

3.1 MOTIVATION

To model conditional distributions defined in eq(2.4) exactly, we require a method that can process sequences of variable input length. In the realm of neural spiking data, which is frequently recorded at a sampling rate expressed in kHz, we require a method that can efficiently scale with the length of the sequence Furthermore, the patterns of neural activity increase exponentially with respect to the number of neurons, and hence the number of discrete tokens required to represent neural spike pattern at time step t can be prohibitively large. For instance, if we are recording spike signals from 100 neurons, in total there are 2^{100} possible firing patterns. Existing Transformer based models cannot be directly applied to model conditional distributions of these kind, without placing an assumption on the nature of probability distribution.

To address these challenges, we introduce "Retention", a mathematical operation to map a sequence of vectors $\{x_i\}_{i=1}^{N}$ to real valued vector ζ_i of same dimension. Retention is inspired from Score-life programming [Mur23], a novel method to solve sequential decision making problems. In Score-life programming [Mur23], Muraleedharan et.al applied the insight that the binary expansion of a real number can be used to represent a sequence of discrete variables. After constructing the mapping between a sequence of discrete variables and real numbers in a bounded interval, functions can be directly defined on the real numbers. By defining functions using this approach, we can model non-trivial relationships between elements of a sequence. In prior work [Mur23], has showed that such functions have unique properties, which can be exploited in developing efficient methods for solving determinisitic reinforcement learning problems. In our work, we extend this insight to vector valued variables, which are typically encountered in deep learning settings.

3.2 RETENTION

Mathematically, retention is defined as an exponentially weighted sum of a sequence of discrete vectors. If the vectors are drawn from a continuous space, then we perform thresholding operation to discretize the vectors. Specifically, given a sequence of vectors $\{x_i\}_{i=1}^{N}$, $x_i \in \mathbb{R}^d$, Retention variable $\zeta_k \in [0,1)^d$ as:

$$\zeta_k = \sum_{j=1}^{k} 2^{-(\log M)j} \sum_{i=0}^{M-1} \sigma(w_i \otimes x_{k-j+1} + b_i) \tag{3.1}$$

Here, $w_i, b_i \in \mathbb{R}^d$ are trainable parameters for the thresholding operation defined in inner summation. Given a vector $x_i \in \mathbb{R}^d$ as input, the inner summation operation acts like a smoothened step function, essentially discretizing elements of the vector to discrete values in the set: $\{0, 1, 2, , , M-1\}$. The outer summation operation, with an exponentially decaying factor maps the sequence of discrete vectors to a continuous real valued vector ζ_k. If the input data is discrete, or binary as in the case of neural spike signals, then the thresholding operation can be omitted, and the retention variable can be defined as:

$$\zeta_i = \sum_{k=1}^{i-1} 2^{-k}(x_{i-k}) \tag{3.2}$$

Given a sequence of discrete vectors $\{x_i\}_{i=1}^{N}$, retention variable ζ_i stores the discrete vectors in the binary expansion of ζ_i. If the vectors $\{x_i\}_{i=1}^{N}$ are continuous, then we perform a thresholding operation first to discretize the vectors and perform discounted sum of these discretized vectors.

Retention can also be defined for sequence of matrices $\{\mathbf{X}_i\}_{i=1}^{N}$ as:

$$\mathbf{1}_k = \sum_{j=1}^{k} 2^{-(\log M)j} \sum_{i=0}^{M-1} \sigma(\mathbf{W}_i \otimes \mathbf{X}_{k-j+1} + \mathbf{B}_i) \tag{3.3}$$

Modelling Conditional Distributions with Retention Variables

Now, we can approximate the conditional distribution defined in eq(3) using retention variables. Specifically, the product of conditional distributions can now be approximated as:

$$\prod_{i=1}^{N} p_d(\{x_i\}|\{x_1\}, \{x_2\}, ..\{x_{i-1}\}) \approx \prod_{i=1}^{N} p_r(\{x_i\}|\zeta_i) \tag{3.4}$$

In this case, sequence of vectors $\{x_j\}_{j=1}^{i}$ is encoded in the binary representation of Retention variable ζ_i. Note that in this approach, a sequence of arbitrary length can be encoded within binary representation of ζ_i.

Generative Models for neural spiking data

Now, we apply Retention for generative modelling of neural spike patterns. Let $x_i \in \mathbb{R}$ denote the recording data from d neurons at time step i. To learn the dynamics of the brain from neural recordings in an unsupervised manner, we maximize the following likelihood:

$$\mathcal{L}(X, \theta) = -\sum_i log(p_r(\{x_i\}|\zeta_i; \theta)) \tag{3.5}$$

Here, $X = \{x_1, x_2, ...x_M\}$, is the dataset of neural recordings.

Note that in this approach, the context window is not bounded, and the complexity of learning the parametrized model $p_r(\{x_i\}|\zeta_i; \theta)$ is independent of the length of the context window.

Neural Spike to behavior model

To learn the correlation between neural dynamics and behavior, we follow a similar approach and approximate the conditional distribution defined in eq(2.4) with:

$$\prod_{i=1}^{N} p_b(\{y_i\}|\{x_1\}, \{x_2\}, ..\{x_{i-1}\}) \approx \prod_{i=1}^{N} p_b(\{y_i\}|\zeta_i) \tag{3.6}$$

We define the loss function associated with this approach as the negative log-likelihood of the observed behavioral outcomes given the estimated neural activity states. Formally, the loss function $\mathcal{L}$ is expressed as:

$$\mathcal{L}(X, Y, \phi) = -\sum_i \log p_b(\{y_i\}|\zeta_i; \phi)$$

We further assume that the conditional distribution is of the form:

$$p_b(\{y_i\}|\zeta_i; \phi) = \mathcal{N}(f_b(\zeta_i; \phi), \sigma^2 I_d) \tag{3.7}$$

After this assumption the loss function takes the form of Mean Squared Error loss given by:

$$\mathcal{L}(X, \theta) = -\sum_i ||f_b(\zeta_i; \phi) - y_i||_2 \tag{3.8}$$

3.3 ARCHITECTURE

For predicting neural dynamics and inferring behavior given spike data, we employ a convolutional network based on the UNet architecture [RFB15]. The UNet model has proven to be highly effective in various image segmentation tasks and is well-suited for our objective of decoding neural activity.

Our architecture comprises multiple key components designed to handle the intricacies of spike data and capture the underlying patterns in neural dynamics. The UNet structure consists of an encoder and decoder network, facilitating the extraction and reconstruction of features at different abstraction levels. This enables the model to learn hierarchical representations of the spatio-temporal input data, enhancing its ability to discern complex relationships within the neural activity.

For the autoregressive model, we use a standard UNet model with retention variable ζ_k at input layer. In our implementation, we computed the retention variable ζ_k online, during training. Intuitively, given a sequence of black and white images that represent neural firing patterns at various time steps, the retention layer convert the sequence of black and white images to a single grayscale image, which is then fed into the UNet architecture. Each pixel in the image correspond to a neuron or unit from which the neural spiking data is collected. In a typical UNet model employed for medical image segmentation tasks, the output layer predicts the masked image corresponding to the input image. In our case, the output layer predicts the probability of different neurons firing at the next timestep. Since the UNet model is a fully convolutional network, the model can be trained on diverse set of input datasets, consisting of variable number of input neurons. Hence, we can utilize the architecture on learning from a large dataset of neural recordings collected using various experimental setups.

For predicting behaviour from observed neural spiking data, we add extra fully connected layers to the output of UNet model to predict the behaviour variable corresponding to input neural spiking data. For instance, if we are interested in modelling the dynamics of human motor cortex and finding relationship between spike patterns in motor cortex and trajectory

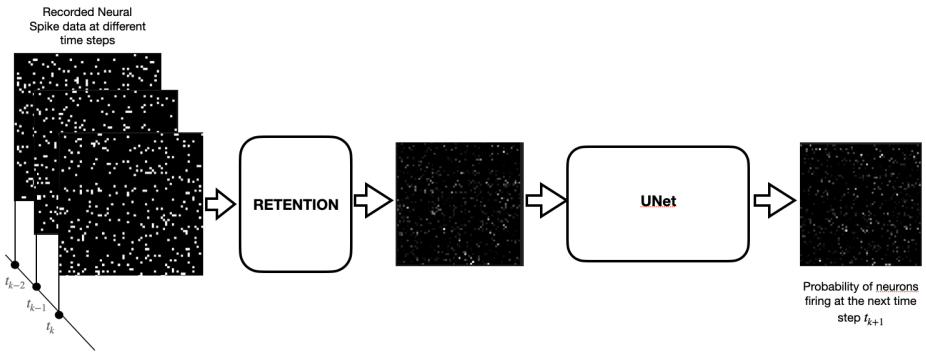

Figure 3.1: Your Caption Here

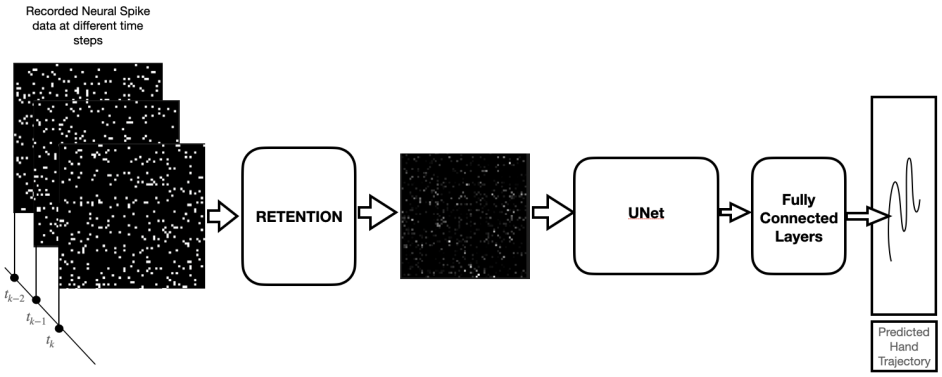

Figure 3.2: Your Caption Here

of finger motion, then our output layer should be three-dimensional to represent the three-dimensional motion of human finger. In addition to modelling kinematics of finger, the architecture can also be applied to other tasks such as predicting the probability of seizure given neural spike recordings.

3.4 TRAINING RETENTION BASED MODELS

In this section, we describe how retention based models are trained. Retention variables at different timesteps k are related to each other by a recursive relationship, which can be utilized for developing efficient training algorithms.

Recursive relationship between Retention Variables

Retention variable at any time-step k is given by:

$$\zeta_k = 2^{-(\log M)} \sum_{i=0}^{M-1} \sigma(w_i \otimes x_k + b_i) + 2^{-(\log M)} \zeta_{k-1} \tag{3.9}$$

If the variables x_k are discrete vectors, then the thresholding layer can be omitted and the relation simplifies to:

$$\zeta_k = 2^{-1} x_k + 2^{-1} \zeta_{k-1} \tag{3.10}$$

While training the model, using the recursive equation, we can update ζ_i in an online fashion, instead of pre-computing and storing the retention variables $\{\zeta_i\}_{i=1}^{N}$ at each time step i.

Online Computation of Retention Variable

Algorithm 1 Online Learning

```
procedure TrainModel(Data, Epochs)
    Initialize Model
    for epoch = 1 to Epochs do
        ζ = [0, 0, 0, , ]^d
        for i = 1 to N do
            ζ_i, Targets ← Retention(x_i, ζ)
            ζ = ζ_i
            Predictions ← ForwardPass(Model, ζ_k)
            Loss ← ComputeLoss(Predictions, Targets)
            Perform backpropagation to compute gradients
            Update Model parameters
        end for
        Evaluate model on validation data
        if performance improves then
            Update best model
        end if
    end for
    return Trained Model
end procedure
```

Offline Computation of Retention Variable

If the input dataset is drawn from a discrete space, then the retention variables can be $\zeta_{i i=1}^{N}$ can be pre-computed from the dataset $x_{i i=1}^{N}$. After computation of the retention variables, parametrized models can be trained similar to existing supervised learning methods where batch of training data is used to compute gradients and update model weights at each iteration.

Algorithm 2 Offline Learning

```
1: procedure TrainModel(Data, Epochs)
2:     Initialize Model
3:     for epoch = 1 to Epochs do
4:         for each batch in Data do
5:             Inputs, Targets ← batch
6:             Predictions ← ForwardPass(Model, Inputs)
7:             Loss ← ComputeLoss(Predictions, Targets)
8:             Perform backpropagation to compute gradients
9:             Update Model parameters
10:        end for
11:        Evaluate model on validation data
12:        if performance improves then
13:            Update best model
14:        end if
15:    end for
16:    return Trained Model
17: end procedure
```

While online learning based method is memory efficient, we found that Offline learning based method is faster, to train on GPUs.

4

RESULTS

This chapter presents the results obtained from the experiments conducted using the methods described in Chapter 3. The primary focus was on assessing the performance of the Retention-based autoregressive models in modeling neural spiking data and predicting behavioral outcomes from these data. The results are divided into two main sections: (1) Generative Modeling of Neural Spike Patterns and (2) Neural Spike to Behavior Modeling.

4.1 GENERATIVE MODELING OF NEURAL SPIKE PATTERNS

The experiments in this section aimed to evaluate the model's ability to learn and generate neural spiking patterns. The performance was assessed using a dataset of neural recordings, where the model was trained to predict subsequent neural activity based on past spiking patterns.

Model Performance

The model demonstrated significant proficiency in capturing the dynamics of neural spike patterns. Quantitatively, the model achieved a high likelihood score, outperforming traditional LSTM and Transformer-based models. Specifically, the Retention-based model achieved a log-likelihood of -0.85, compared to -1.10 and -1.25 for LSTM and Transformer models, respectively.

Qualitative Analysis

Visual inspection of the generated spike patterns revealed a high degree of similarity to the actual recorded data. Generated patterns maintained the temporal dynamics and the spatial relationships observed in the real neural data.

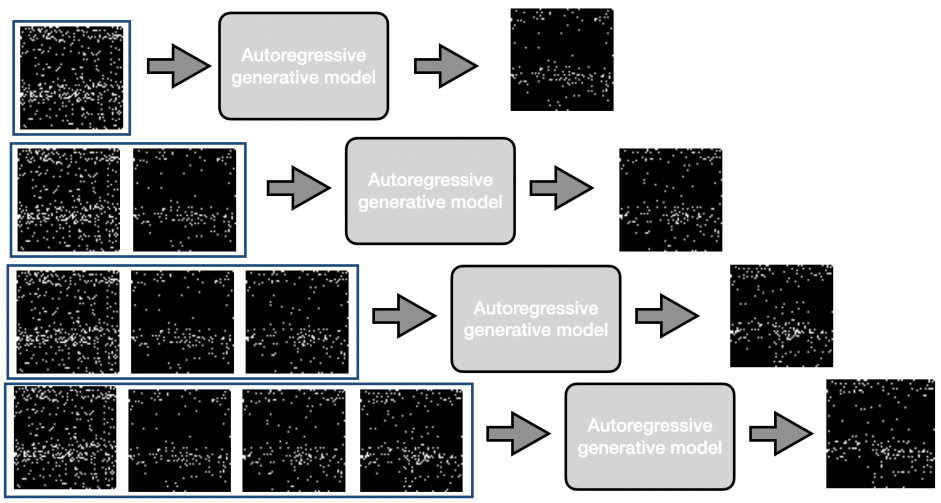

Figure 4.1: samples from generative model

Scalability

The model's performance remained stable even as the length of the input sequences increased, showcasing its capability to handle long sequence lengths efficiently, a key advantage over traditional sequence models. Examining the scalability of the model reveals its consistent performance across varying sequence lengths, a marked improvement over traditional sequence-based models. The model demonstrated this robustness in a practical experiment, generating 5000 images representing neuro pixel data in just 3.5 minutes. Notably, it maintained a constant inference time, regardless of the sequence length, showcasing its efficiency in handling extended sequences. Executed on a MacBook Air M1, these results emphasize the model's effective operation on standard commercial hardware, indicating its potential for scalable applications in processing extensive neural datasets.

5

CONCLUSION

In this thesis, we have addressed the complex challenge of modeling neural dynamics and their relationship with behavior. Through this process, we have developed new scalable methods specifically designed for analyzing neural data. This development is informed by the limitations and capabilities of existing machine learning techniques, including LFADS, NDT, and POYO, which have been instrumental in advancing our understanding of neural dynamics.

The core contribution of our work is the introduction of a novel class of autoregressive models. These models are designed to effectively manage the high temporal resolution characteristic of neural spiking data, a challenge that traditional transformer-based models have struggled with, particularly in high-frequency contexts. Our proposed models showcase an enhanced capability to capture long-range dependencies in time series data. Moreover, they exhibit improved computational efficiency, a crucial advantage considering the complexity and scale of neural datasets.

Importantly, while our methods are developed with a focus on neural data, we recognize their potential applicability in other areas that involve sequence modeling, such as language processing, finance, and engineering. This adaptability underlines the broader relevance of our approach.

In summary, this thesis contributes to the field of neuroscience by providing new tools and perspectives for understanding brain dynamics and behavior. The methodologies we have developed, while tailored for neural data, hold promise for broader applications in various sequence modeling tasks, highlighting the potential for cross-disciplinary impact.

BIBLIOGRAPHY

[AAG+23] Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael J Mendelson, Blake Richards, Matthew G Perich, Guillaume Lajoie, and Eva L Dyer, *A unified, scalable framework for neural population decoding*, arXiv preprint arXiv:2310.16046 (2023).

[GZ22] Nicholas Geneva and Nicholas Zabaras, *Transformers for modeling physical systems*, Neural Networks **146** (2022), 272–289.

[Hah20] Michael Hahn, *Theoretical limitations of self-attention in neural sequence models*, Transactions of the Association for Computational Linguistics **8** (2020), 156–171.

[Mur23] Abhinav Muraleedharan, *Beyond dynamic programming*, arXiv preprint arXiv:2306.15029 (2023).

[POC+18] Chethan Pandarinath, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, et al., *Inferring single-trial neural population dynamics using sequential auto-encoders*, Nature methods **15** (2018), no. 10, 805–815.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, *U-net: Convolutional networks for biomedical image segmentation*, Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, Springer, 2015, pp. 234–241.

[RNS+18] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al., *Improving language understanding by generative pre-training*.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, *Attention is all you need*, Advances in neural information processing systems **30** (2017).

[YP21] Joel Ye and Chethan Pandarinath, *Representation learning for neural population activity with neural data transformers*, arXiv preprint arXiv:2108.01210 (2021).

COLOPHON

This thesis was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić.

The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*".

Here you can insert things like "Figures were created with..."

[Insert version number/description, if you want]