Topics in Generalization and Learnability of Modern Machine Learning

by

Jacob Trauger

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in The University of Michigan
2026

Doctoral Committee:

Professor Ambuj Tewari, Chair
Assistant Professor Saptarshi Chakraborty
Professor Clayton Scott
Associate Professor Yuekai Sun

Jacob Trauger
jtrauger@umich.edu
ORCID iD: 0009-0008-3590-8832

DEDICATION

To quote a song:

This is for the lions living in the wiry broke down frames of my friends bodies... This is for the snakes and the people they bite, for the friends I've made, for the sleepless nights, for the warning signs I've completely ignored...

ACKNOWLEDGEMENTS

Everyone in this acknowledgments section deserves way more words than I will write below about them. However, in keeping with traditions of many of the most important and well-used textbooks, the extra words are left as exercise for the reader to ask me about.

I would first like to thank my advisor Ambuj Tewari. Throughout the years, your consistent enthusiasm, excitement, and passion for our research has inspired me to continue during the rollercoaster that is research. I truly appreciate all the guidance as well, without it I would not be where I am now. I would like to thank my committee: Clayton Scott, Yuekai Sun, and Saptarshi Chakraborty, for your invaluable time and effort as well.

To my family: Mom, Dad, Jenny, all my aunts, uncles, grandparents, cousins, and more, thank you for supporting me and helping me to grow into the person I am today. To my siblings: Maryn, Molly, Ty, and Will[1], you guys are awesome and I love you all so much. Thank you for being so cool. You can not choose your family, but I would choose you all in a heartbeat.

The family I did choose deserves acknowledgments as much as everyone else. Extremely extremely extremely special thanks to Maya Bose, Erin Holcomb, Brandon Dengel, and Alyssa McDonald for being my best friends over the years. You guys have been there for me no matter what and I will always appreciate the fantastic times we have had (and will continue to have) together. To my high school friends Corey Tonachio and Nick Gerber, I love how we have stayed in contact throughout the years even though we are all thousands of miles from each other. Let's keep that up. To previous residents of 321 East Ann Street: Keagan Moo, Grant Carr, Miguel Limcaoco, Matt Conrad, and Brandt Bessell, thank you all for being my first friends here and I always look forward to whatever antics we get up to together. Don't forget to share the golden cow. To the rest of my friends outside the stats department: Ashley Bielawski, Abe Raskind, Mar Dolorfino, and everyone else, I will always appreciate hanging out, learning, and climbing up the walls with you guys. You all are so cool and I am glad to call you all my friends.

This thesis would also probably be longer if I had no friends in the stats department to distract me[2], but thankfully this is not the case. I would like to say thank you to Jaylin

[1] listed in alphabetical order; no you are not better because you were listed before someone else

[2] they were distracting me and definitely not the other way around

Lowe, Gabe Durham, Marc Brooks, Paolo Borello, Luke Francisco, Chinmaya Kausik, Yash Patel, Vinod Raman, Unique Subedi, Seamus Somerstep, Josh Wasserman, Daniel Zou, Noah Kochanski, Josh Garman, Sahana Rayan, Saptarshi Roy, Joe Pennacchio, and everyone else in the department who has made my time in West Hall a time I will not forget. You guys are also super cool and I am glad to call you all my friends as well.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF APPENDICES

APPENDIX

ABSTRACT

Learning theory is a subfield of machine learning research where we analyze the theoretical properties of machine learning problems and algorithms through mathematics. This thesis is a culmination of three standalone works in the field of learning theory.

First, we analyze the generalization bounds of the Transformer architecture to show that it does not depend on maximum sequence length. To do this, we analyze the Rademacher Complexity of the architecture and create novel covering number bounds on linear functions that do not depend on the amount of samples. We also run a simulation and show the results support our theoretical findings.

In the next chapter, we analyze a quirk seen in the training of modern large language models. Most of these models are trained to only predict the next token of the output; however, the output of the model is a sequence of tokens. We study this mismatch in training optimization and output through the lens of the surrogate loss consistency framework. We analyze different ways of decoding these next-token predictors to see when we achieve asymptotic consistency for two use cases when encoded as loss functions.

In the final work of this thesis, the theoretical learnability of multiclass forgiving 0-1 loss functions is studied through the PAC-Learnability framework. We show a generalization the Natarajan dimension [Natarajan, 1989] characterizes the learnability of many instantiations of learning problems that use forgiving 0-1 loss functions. We also show how this setting can be used to model other known settings in the literature.

CHAPTER 1

Introduction

Machine learning is as large today as it has ever been and the practicality and usefulness of deep learning techniques can not be understated. Moreover, the state of machine learning progress has grown rapidly over the last twenty years and the big data revolution has transformed all of our lives immensely. From large language models to even having NFL commentators discuss a model's prediction of catch probability during a football game, deep learning is all around us. While the practical use cases of these algorithms and models continues to grow, so does the need to ground the theory of the field. This thesis is a small part in that effort.

This thesis explores various avenues of research on the theoretical underpinnings of deep learning. The results from such work allow us to discover insights about how an architecture works, understand practical limitations in real world settings, and even discuss when a problem is learnable or not.

The summary of the chapters and their contributions are as follows:

- In Chapter 2, we study the architecture at the heart of large language models: the Transformer [Vaswani et al., 2017]. Through repeated blocks of self-attention layers composed with dense neural networks, this architecture has revolutionized the way the world works. The Transformer uses sequence data as input and one parameter in the architecture is the maximum sequence length it can handle. In the chapter, we study the generalization bounds of the Transformer architecture (e.g. if we draw a sample from a distribution $\mathcal{D}$, how far off is the sample loss of our algorithm from its expected loss on $\mathcal{D}$) and we show that these high probability bounds do not depend on the maximum sequence length.

- In Chapter 3, we study a different part of the large language models paradigm. Most large language models are trained to, given a context, predict only the next token. Then, a decoding algorithm chooses the next token and this process repeats the until done. Thus, these next-token predictors are the object that is minimized through

training, however, what the end user cares about is the resulting output sequence. This mismatch in what is being optimized over versus what is important is the dichotomy we study. We use the surrogate loss consistency framework to study when minimizing the loss on the next-token predictors optimizes the sequence outputs for various user end goals encoded as loss functions. We find that, depending on whether the user desires correctness versus creative generation, the decoding algorithms that optimize the output sequence changes. This has been seen empirically in the recent literature on decoding algorithms, but has never had firm theoretical grounding until now as far as we are aware.

- In Chapter 4, we study when specific learning problems are even learnable. In a multiclass classification scenario, we study the cases when the loss functions have a range of $\{0, 1\}$ and allow for multiple outputs to have 0 loss for multiple labels. We call these "forgiving" 0-1 loss functions. We use the Probability Approximately Correct (PAC) framework to show that many instantiations of these learning problems are characterized by a generalization of the Natajaran dimension.
- In Chapter 5, we conclude with final thoughts.

CHAPTER 2

Sequence Length Independent Norm-Based Generalization Bounds for Transformers

In this chapter we study the generalization bounds of the Transformer architecture. When training machine learning models, we train on a training set, but we want to do well on the true distribution. The difference between this training loss and the expected loss over the entire distribution is called the generalization gap. Being able to bound this gap is a useful tool for analyzing an architecture as it gives insights into what model parameters make the model need more samples to learn well.

We will provide a norm-based generalization bound for the Transformer architecture that does not depend on the input sequence length. We employ a covering number based approach to prove our bounds. We use three novel covering number bounds for the function class of bounded linear mappings to upper bound the Rademacher complexity of the Transformer. Furthermore, we show this generalization bound applies to the common Transformer training technique of masking and then predicting the masked word. We also run a simulated study on a sparse majority data set that empirically validates our theoretical findings.

2.1 Introduction

Since Vaswani et al. [2017] debuted the Transformer, it has become one of the most pre-eminent architectures of its time. It has achieved state of the art prediction capabilities in various fields [Dosovitskiy et al., 2020, Wu et al., 2022, Vaswani et al., 2017, Pettersson and Falkman, 2023] and an implementation of it has even passed the BAR exam [Katz et al., 2023]. With such widespread use, the theoretical underpinnings of this architecture are of great interest.

Specifically, this chapter is concerned with bounding the generalization gap when using the Transformer in supervised learning. These upper bounds can be used to help understand how sample size needs to scale with different architecture parameters and they are a very common theoretical tool to understand machine learning algorithms [Kakade et al., 2008,

Garg et al., 2020, Truong, 2022, Lin and Zhang, 2019].

One such architecture parameter is the sequence length of the input. Since the input of Transformers can be thought of as a sequence of tokens (e.g. a sequence of word embeddings), the maximum allowable length of an input sequence is called the sequence length.

The main contribution of this chapter is providing norm-based generalization bounds for the Transformer architecture that have no explicit dependence on the input sequence length. We also contribute 3 novel vector valued linear mapping covering number bounds that are the key to obtain our generalization bounds. Furthermore, we give an example of a regime where our bounds apply and we give empirical evidence that our theory holds.

Previously, the best known norm-based generalization bound scaled with the logarithm of the sequence length [Edelman et al., 2022]. Removing the dependence on sequence length leads to more intuitively appealing bounds since the total number of parameters in the Transformer is independent of input sequence length. Since our bounds are norm-based they have potential to provide meaningful guarantees even in overparameterized regimes where parameter counting bounds might be less meaningful.

We are able to show this by going through the Rademacher complexity of the Transformer and then using three novel linear covering number bounds to bound the Rademacher complexity. Therefore, Section 1 goes over the necessary background needed. Section 2 shows the novel covering number bounds for a linear mapping function class with bounded matrices and inputs. In Section 3 we start dealing specifically with Transformers. Here we show a new Rademacher complexity bound for a single layer Transformer. Section 4 provides details on how our covering number bounds can be used in the multi-layer analysis of Edelman et al. [2022] to get a sequence length independent norm-based generalization bound. Section 5 shows how a method of training used in BERT [Devlin et al., 2018] can be reduced to what is studied in this chapter. Finally, in Section 7 we show an experiment on a simulated sparse majority data set to empirically validate our theoretical findings.

2.1.1 Related Works

This chapter is most closely related to the work of Edelman et al. [2022], whom prove a norm-based generalization bound that grows logarithmically with sequence length. Due to this, they state Transformers have an inductive bias to represent a sparse function of the inputs. We bolster this claim further by removing the dependence on sequence length altogether.

Another result similar to our is given by Zhang et al. [2022]. They are able to remove the dependence on sequence length. However, the bound they get, which we shall call a parameter counting-based bound, has several drawbacks which we discuss in Section 2.2.5.

Wei et al. [2022] also show generalization bounds for Transformers, but specifically study binary classification setting with 0-1 loss and use a margin approach. Fu et al. [2023] freeze some of the weight matrices at initialization and bound the excess risk in this setting as a function of the amount of heads in the attention layer. This chapter's bound also do not depend on sequence length.

Outside of Transformers, using Rademacher complexity in deep learning to bound the generalization gap has a rich history. Golowich et al. [2018] was able to use Rademacher complexities to get a generalization bound independent of the depth and width of a neural network. Truong [2022] is able to use Rademacher complexity to get nearly tight bounds on neural networks under some assumptions on the data. Also, Bartlett et al. [2017] use covering numbers and Rademacher complexity to get generalization bounds on multiclass neural networks using a margin based approach.

2.2 Background

2.2.1 Matrix Definitions

For our matrix notation we will, in general, use capital letters for matrices and lowercase letters for vectors. For a matrix W, we will use $W_{:,i}$ to denote the i^{th} column of W and W_i to denote the i^{th} row unless otherwise noted.

Now, we will define a few well-used matrix norms. Let $p, q, r, d \in \mathbb{N}$ and let $W \in \mathbb{R}^{r \times d}$. The first norm, denoted as $\|W\|_{q,p}$, will be defined as $\|W\|_{q,p} = \left\| [\|W_{:,1}\|_q, \dots \|W_{:,d}\|_q]^\top \right\|_p$.

Another matrix norm, also known as the operator norm, we will denote as $\|W\|_{q\to p}$. This one is defined as:

$$\|W\|_{q\to p} = \sup_{x \in \mathbb{R}^d \setminus 0} \frac{\|Wx\|_p}{\|x\|_q}$$

One final matrix norm we will review is the Frobenius norm, denoted as $\|W\|_F$, which is defined as:

$$\|W\|_F = \sqrt{\sum_{i=1}^{r} \sum_{j=1}^{d} W_{ij}^2}$$

We will also denote $\|W\|_{2\to 2}$ as $\|W\|_2$. A well known property of the $\|\cdot\|_{2\to 2}$ operator is that it is equal to the largest singular value of the input matrix. The Frobenius norm is also well known to be equal to the the square root of the squared sum of the singular values of a matrix. Therefore, we have $\|W\|_{2\to 2} \leq \|W\|_F$ for any matrix W.

2.2.2 Generalization Bounds

When training machine learning algorithms, we can only use a finite amount of data to learn from, however, we want our resulting function to generalize well outside of our training sample. Thus, having guarantees with high probability on the difference between the loss on our training sample and the loss on our testing population is extremely important. Generalization bounds try to upper bound this loss gap.

Mathematically, if we have a hypothesis class $\mathcal{H}$, sample space $\mathcal{X}$, label space $\mathcal{Y}$, loss function ℓ, and distribution over the sample and label space $\mathcal{D}$, then our generalization gap for a set of samples and labels $S = \{(x_i, y_i)\}_{i=1}^n$, $x_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$, on the hypothesis $h \in \mathcal{H}$ is defined to be

$$\left| \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell(h(x), y) \right] - \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) \right|$$

Notice how if we can have this value go to 0 with high probability over all sets of samples and for all $h \in \mathcal{H}$, then we can be confident that minimizing the sample loss will not impact our generalization.

2.2.3 Rademacher Complexity

One such tool that can be used to upper bound the generalization gap is the Rademacher complexity. Let us have the same set up as in the previous section. Then the Rademacher complexity of a hypothesis class $\mathcal{H}$ is defined to be

$$Rad_n(\mathcal{H}, S) = \frac{1}{n} \mathbb{E}_\sigma \left[\sup_{h\in\mathcal{H}} \sum_{i=1}^{n} \sigma_i h(x_i) \right]$$

where each σ_i are i.i.d. and take values ± 1 each with half probability and $\sigma = (\sigma_1, \ldots, \sigma_n)$. It is well known that [Shalev-Shwartz and Ben-David, 2014a], if the magnitude of our loss function is bounded above by c, with probability greater than $1 - \delta$ for all $h \in \mathcal{H}$, we have

$$\left| \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell(h(x), y) \right] - \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i) \right| \leq$$
$$2Rad_n(\ell \circ \mathcal{H}, S) + 4c\sqrt{\frac{2\log(4/\delta)}{m}}$$

where $\ell \circ \mathcal{H} = \{(x, y) \mapsto \ell(h(x), y) \mid h \in \mathcal{H}\}$. Therefore, if we have an upper bound on the Rademacher complexity, we can have an upper bound on the generalization gap.

2.2.4 Covering Numbers

The use of covering numbers is one such way we can bound the Rademacher complexity of a hypothesis class. Let $q \in \mathbb{R}_{>0}$ and let us have a function class $\mathcal{F}$, $\forall f \in \mathcal{F}\ f : \mathbb{R}^d \to \mathbb{R}^k$. We say a subset $\hat{\mathcal{F}} \subset \mathcal{F}$ covers a set of inputs $\{x_i\}_{i=1}^n$ if for every $f \in \mathcal{F}$, $\exists \hat{f} \in \hat{\mathcal{F}}$ such that $\sup_{x_i} \left\| f(x_i) - \hat{f}(x_i) \right\|_q < \epsilon$. We will use the notation $N_\infty(\mathcal{F}, \epsilon, \{x_i\}_{i=1}^n, \|\cdot\|_q)$ for this. We will define the covering number, $N_\infty(\mathcal{F}, \epsilon, n, \|\cdot\|_q)$ as

$$\sup_{\{x_i\}_{i=1}^n} N_\infty(\mathcal{F}, \epsilon, \{x_i\}_{i=1}^n, \|\cdot\|_q)$$

It has been shown that for scalar valued hypothesis classes, the Rademacher complexity can be upper bounded using the covering number of the hypothesis class [Dudley, 1967]. Using a slightly modified version, we have for a constant c:

$$Rad_n(f, S) \leq c \inf_{\delta \geq 0} \left(\delta + \int_\delta^\infty \sqrt{\frac{\log N_\infty(\mathcal{F}, \epsilon, n)}{n}} d\epsilon \right)$$

Notice that if we have a bounded function class, then the ∞ in the integral limit can become the upper bound of the function class.

2.2.5 Two Types of Bounds

Suppose our inputs have dimension d and the inputs have an upper norm bound of B_x. Suppose our matrices have dimension $k \times d$ and have an upper norm bound of B_w. Here we will note two different types of generalization bounds that can one can arrive at depending on the perspective. One is where you have the norm bounds inside a log term and the dimensions on the outside (e.g., $O(\sqrt{dk \log(B_x B_w))/n})$. The other is where you have the bounds outside the log term and parameters inside (e.g., $O(\sqrt{B_x B_w \log(dk))/n})$. We will refer to the former as parameter-counting type bounds and the latter as the norm-based type bounds. We note that these are not the only types of bounds, just these are the ones relevant to this chapter.

Notice how, for parameter counting bounds, over-parameterized architectures can lead to vacuous bounds. Also notice that parameter counting bounds do not take full advantage of SGD's implicit regularization since the norms are within the logarithm. In contrast, as long as the norm bounds are reasonable, norm-based generalization bounds work well with over-parameterized architectures and work well with implicit regularization.

For Transformers, there are norm-based bounds that scale logarithmically with sequence

length [Edelman et al., 2022] and parameter counting bounds that do not scale with sequence length [Zhang et al., 2022]. This is a gap in the literature which this chapter intends to fill.

2.2.6 Self-attention and Transformers

We will follow the definition of self-attention and Transformers set forth by Edelman et al. [2022] and keep with their notation.

Let $W_c \in \mathbb{R}^{k\times d}$, $W_v \in \mathbb{R}^{d\times k}$, and $W_Q, W_K \in \mathbb{R}^{d\times T}$ be trainable weight matrices. Let $X \in \mathbb{R}^{T\times d}$ be the input, which can be thought of as sequence of T d-dimensional tokens. Also, let σ be a L_σ-Lipshitz activation function that is applied elementwise and has the property $\sigma(0) = 0$. Finally, let RowSoftmax donote applying softmax on each row of its input. Then, they define a Transformer head as

$$\sigma\left(\text{RowSoftmax}\left(XW_QW_K^\top X^\top\right)XW_v\right)W_c$$

Since W_Q and W_K are only ever multiplied with each other, we will combine $W_QW_K^\top$ into a single matrix $W_{QK} \in \mathbb{R}^{d\times d}$ for convenience of analysis. Once we do this, note that the total dimensionality (of W_{QK}, W_c, W_v) is independent of T, the sequence length which counts how many tokens are in each sample. The embedding dimension is d since it is the dimension that the values in the sequence are embedded into and k is the hidden dimension.

For multi-head Transformers, we assume each head is summed up at the end of each layer. That is, the output for a layer of a multi-head Transformer is:

$$\sum_{h=1}^{H}\sigma\left(\text{RowSoftmax}\left(XW_{h,Q}W_{h,K}^\top X^\top\right)XW_{h,v}\right)W_{h,c}$$

Note how the ouput of a layer can be used as the input to another layer. This is how multi-layer Transformer networks are created. Standard practice is to add layer normalization in between each layer as this has been well studied to aid in optimization and generalization [Ba et al., 2016, Wang et al., 2019, Xu et al., 2019]. Thus, keeping with the definitions and notation previously set forth, we will inductively define an L-layer Transformer block:

Let $\mathcal{W}^{(i)} = \{W_v^{(i)}, W_c^{(i)}, W_{QK}^{(i)}\}$ and let $\mathcal{W}^{1:i} = \{\mathcal{W}^1, \ldots \mathcal{W}^{(i-1)}\}$. Also, let

$$f(X; W^{(i)}) = \text{RowSoftmax}\left(XW_Q^{(i)}W_K^{(i)\top}X^\top\right)XW_v^{(i)}$$

$$g_{block}^1\left(X; W^{1:1}\right) = X$$

Then, the output of the i^{th} layer is defined to be

$$g_{block}^{(i+1)}\left(X; W^{1:i+1}\right) = \\ \Pi_{norm}\left(\sigma\left(\Pi_{norm}\left(f\left(g_{block}^{(i)}\left(X; W^{1:i}\right); W^{(i)}\right)\right)\right) W_c^{(i)}\right)$$

where Π_{norm} projects each row onto the unit ball.

In our analysis we will focus on the scalar output setting for Transformers. Specifically, we will follow how BERT trains for scalar output [Devlin et al., 2018]. In order to get a scalar output, we add an extra input in the sequence that can be constant or trained. Let this index be the $[CLS]$ index. Let us also have a trainable vector $w \in \mathbb{R}^d$. Then, at the last layer, take the output at the $[CLS]$ index, $Y_{[CLS]} \in \mathbb{R}^d$ and multiply it with w to get our output $w^\top Y_{[CLS]} \in \mathbb{R}$.

2.3 Linear Covering Number Bounds

In this section we will show three different covering number bounds for linear function classes with different restrictions on input and matrix norms. To show the first one, we need the following lemmas, the first one is attributed to Maurey [Pisier, 1981] and first used in this context by Zhang [2002]:

Lemma 2.3.1. *(Maurey's Sparsification Lemma) Let $\mathcal{H}$ be a Hilbert space and let each $f \in \mathcal{H}$ have the representation $f = \sum_{i=1}^d \alpha_i V_i$ where $V_i \in \mathcal{H}$, $\|V_i\| \leq b$ and $\alpha_i \geq 0$ with $\gamma = \|\alpha\|_1 \leq 1$. Then, for any $k \in \mathbb{N}$, there exist $k_1, \ldots, k_d$, $k_i \in \mathbb{Z}_{\geq 0}$, $\sum_{i=1}^d k_i \leq k$, such that*

$$\left\| f - \frac{1}{k} \sum_{i=1}^d k_i V_i \right\|_2^2 \leq \frac{\gamma^2 b^2 - \|f\|^2}{k}$$

We note that the total amount of $(k_1, \ldots k_d)$'s that fit the criteria above is less than or equal to d^k. This has been used to upper bound the covering number for linear functions [Zhang, 2002, Bartlett et al., 2017] and we will use it similarly in our proofs.

For covering a class of scalar valued linear functions $\{x \to w^\top x \mid w \in \mathbb{R}^d$,w norm bounded$\}$ with inputs $\{x_i \in \mathbb{R}^d\}_{i=1}^n$, it is known that we are able to remove the dependence on n and replace it with a dependence on d. Kontorovich and Attias [2021] show this and attribute it to folklore. Given a cover such as this, it is an immediate extension to create a non-n-dependent cover of the function class $\{x \to Wx, W \in \mathcal{W}\}$ as long as each row in each $W \in \mathcal{W}$ is bounded by the norm restraint needed for the scalar valued cover (specific details are in appendix section A.1).

Our first contribution generalizes the above by allowing us to consider more possible norm bounds on $\mathcal{W}$. It shows that, under certain norm restrictions for our input, the linear mappings covering number for any set of size N is equivalent to the covering number on the appropriately scaled standard basis (proof in Appendix Section A.2).

Lemma 2.3.2. *Let $\mathcal{W} \subset \mathbb{R}^{k\times d}$ and let $\mathcal{F} = \{x \to Wx \mid W \in \mathcal{W}\}$ with $\|x\|_1 \leq B_x$. Then, for $N \geq d$, we have*

$$\mathcal{N}_\infty\left(\mathcal{F}, \epsilon, N, \|\cdot\|_q\right) = \mathcal{N}_\infty\left(\mathcal{F}, \epsilon, \{B_x e_1, \ldots, B_x e_d\}, \|\cdot\|_q\right)$$

2.3.1 Log Covering Number for $\|\cdot\|_1$ Bounded Input and $\|\cdot\|_{1,\infty}$ Bounded Matrix

Now, we will show our three covering number bounds. Each of the three have different norm bounds on the inputs and the matrices, allowing for flexibility when deciding which to use.

Lemma 2.3.3. *Let $N \geq d$, $\mathcal{W} = \{W \in \mathbb{R}^{k\times d} \mid \|W\|_{1,\infty} \leq B_w\}$, $\mathcal{F} = \{x \to Wx \mid W \in \mathcal{W}\}$, and let our inputs $x \in \mathbb{R}^d$ have the restriction $\|x\|_1 \leq B_x$. Then:*

$$\log \mathcal{N}_\infty\left(\mathcal{F}, \epsilon, N, \|\cdot\|_2\right) \leq \frac{dB_w^2 B_x^2}{\epsilon^2} \log(2k+1)$$

Proof. We will abuse notation slightly by referring to $\mathcal{W}$ at times instead of $\mathcal{F}$ (and similar for $\hat{\mathcal{W}}$ and $\hat{\mathcal{F}}$).

Let us have the set $V = \{v \in \mathbb{R}^k \mid \|v\|_1 \leq B_w\}$. Then notice by lemma 2.3.1 we have that there exists an ϵ/B_x cover for V that has log size

$$\frac{B_w^2 B_x^2}{\epsilon^2} \log(2k+1)$$

Let this cover be $\hat{V}$. We claim that

$$\hat{\mathcal{W}} = \underbrace{\hat{V} \otimes \hat{V} \cdots \otimes \hat{V}}_{d \text{ total times}}$$

is a cover for $\mathcal{W}$.

To show this, let $W \in \mathcal{W}$ and let $\hat{W} \in \hat{\mathcal{W}}$ be the one where each column is the vector that would be chosen to cover the corresponding column in W. Then, notice for all $i \in [d]$ we have

$$\left\|(W - \hat{W})B_x e_i\right\| = B_x \left\|W_{:,i} - \hat{W}_{:,i}\right\| \leq B_x \frac{\epsilon}{B_x} = \epsilon$$

Therefore

$$\sup_{i\in[d]} \left\|(W-\hat{W})B_x e_i\right\| \leq \epsilon$$

which, by lemma 2.3.2 shows that

$$\log \mathcal{N}_\infty \left(\mathcal{F}, \epsilon, N, \|\cdot\|_2\right) \leq \frac{dB_w^2 B_x^2}{\epsilon^2} \log(2k+1)$$

□

2.3.2 Log Covering Number for $\|\cdot\|_1$ Bounded Input and $\|\cdot\|_{2,1}$ Bounded Matrix

Lemma 2.3.4. *Let $N > d$, $\mathcal{W} = \{W \in \mathbb{R}^{k\times d} \mid \|W\|_{2,1} \leq B_w\}$, $\mathcal{F} = \{x \to Wx \mid W \in \mathcal{W}\}$, and let our inputs $x \in \mathbb{R}^d$ have the restriction $\|x\|_1 \leq B_x$. Then:*

$$\log \mathcal{N}_\infty \left(\mathcal{F}, \epsilon, N, \|\cdot\|_2\right) \lesssim \frac{B_w^2 B_x^2}{\epsilon^2} \log(dk)$$

where $\lesssim$ hides logarithmic dependencies except T, k, d.

Proof. This proof uses Lemma 4.6 in Edelman et al., 2022 Edelman et al. [2022], which is rewritten below for clarity.

Lemma 2.3.5. *(Edelman et al., 2022 Lemma 4.6) Let $\mathcal{W}$ and $\mathcal{F}$ be as above. Then for any set of points $x_1, \ldots, x_n \in R^d$ with $\|x_i\|_2 \leq B_x$ for all i, we have*

$$\log \mathcal{N}_\infty \left(\mathcal{F}, \epsilon, \{x_i\}_{i=1}^n, \|\cdot\|_2\right) \lesssim \frac{B_w^2 B_x^2}{\epsilon^2} \log(dn)$$

With this, let $\hat{\mathcal{W}}$ be an ϵ-cover for $\mathcal{W}$ over the inputs $\{B_x e_i\}_{i=1}^d$ as stated in the lemma. Then, by lemma 2.3.2, we have that the cardinality of $\hat{\mathcal{W}}$ is also an upper bound for $\mathcal{N}_\infty \left(\mathcal{F}, \epsilon, N, \|\cdot\|_2\right)$, which is what we needed to show. □

2.3.3 Log Covering Number for $\|\cdot\|_2$ Bounded Input and $\|\cdot\|_{1,1}$ Bounded Matrix

Lemma 2.3.6. *Let $N \geq d$, $\mathcal{W} = \{W \in \mathbb{R}^{k\times d} \mid \|W\|_{1,1} \leq B_w\}$, $\mathcal{F} = \{x \to Wx \mid W \in \mathcal{W}\}$, and let our inputs $x \in \mathbb{R}^d$ have the restriction $\|x\|_2 \leq B_x$. Then:*

$$\log \mathcal{N}_\infty \left(\mathcal{F}, \epsilon, N, \|\cdot\|_2\right) \leq \frac{B_x^2 B_w^2}{\epsilon^2} \log(2dk+1)$$

Proof. Let $\mathcal{V}$ be the set of all the flatten matrices in $\mathcal{W}$. Note how this implies $\forall v \in \mathcal{V}$, we have $\|v\|_1 \leq B_w$. Then, by Maurey's sparification lemma, we have that there exists an ϵ-cover $\hat{\mathcal{V}}$ of log size at most $\frac{B_w^2}{\epsilon^2}\log(2dk+1)$. We claim if we unflatten $\hat{\mathcal{V}}$ (call this $\hat{\mathcal{W}}$), then $\hat{\mathcal{W}}$ is a $(B_x\epsilon)$-cover for $\mathcal{W}$. Let $W \in \mathcal{W}$, let V be the flatten version of W. Then, let $\hat{V}$ be the flattened vector we would choose for V in our cover and let $\hat{W} \in \hat{\mathcal{W}}$ be the unflattened version of $\hat{V}$. Notice for any $x \in \mathbb{R}^d$, $\|x\|_2 \leq B_x$:

$$\begin{aligned}
&\left\|Wx - \hat{W}x\right\|_2 \leq \left\|W - \hat{W}\right\|_{2\to 2} \|x\|_2 \leq \\
&\left\|W - \hat{W}\right\|_F \|x\|_2 = \left\|V - \hat{V}\right\|_2 \|x\|_2 \leq \\
&\left\|V - \hat{V}\right\|_2 B_x \leq B_x\epsilon
\end{aligned}$$

Therefore our covering number is at most $\frac{B_x^2 B_w^2}{\epsilon^2}\log(2dk+1)$ □

2.3.4 Observations on Results

Above we have showed a few different sharpenings of linear covering numbers with matrices instead of vectors. Specifically, these do not rely on the sample size of the input. Also, all but lemma 2.3.3 keeps the matrix dimensions inside the log term. We do note however, the matrix bound in lemma 2.3.3 is a $1,\infty$ norm bound while the others are rather $2,1$ or $1,1$ norm bound. We know that for any matrix W, $\|W\|_{p,1} \leq d\|W\|_{p,\infty}$. Thus if we were to convert lemmas 2.3.4 and 2.3.6 into a norm bound on $2,\infty$ and $1,\infty$ bounds we would have a $d^2B_w^2$ term. This shows that lemma 2.3.3 is a stronger bound than it lets on.

2.4 Transformer Rademacher Complexity

2.4.1 Analysis for Single Layer Transformer

Let $w \in \mathbb{R}^d$, $W_c \in \mathbb{R}^{k\times d}$, $W_v \in \mathbb{R}^{d\times k}$, $W_{QK} \in \mathbb{R}^{d\times d}$ $\|w\|_1 \leq B_w$, $\left\|W_c^\top\right\|_{1,\infty} \leq B_{W_c}$, and $\left\|W_v^\top\right\|_{1,\infty} \leq B_{W_v}$. Then we have our scalar one layer Transformer as $w^\top Y_{[CLS]}$ where

$$Y_{[CLS]} = W_c^\top \sigma\left(W_v^\top X^\top \text{softmax}\left(XW_{QK}^\top x_{[CLS]}\right)\right)$$

Our Rademacher complexity is thus the following:

$$\mathbb{E}\left[\sup_{w,W_c,W_v,W_{QK}} \sum_{i=1}^m \epsilon_i w^\top W_c^\top \sigma\left(W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)}W_{QK}x_{[CLS]}\right)\right)\right]$$

With this, we have the following theorem:

Theorem 2.4.1. *Suppose we have the required norm restrictions denoted above along with* $\|x\|_2 < B_x \ \forall x \in \mathcal{X}$. *Also, suppose we have a covering number bound in the form of* C/ϵ^2 *for the class* $\{x \to Wx \mid x \in \mathcal{X}, w \in \mathcal{W}\}$ *where* $\mathcal{X}$ *and* W_{QK} *meets these requirements needed as well. Finally, if we have* $m > d$ *and* $m > \ln(2d)$. *Then, an upper bound on the Rademacher complexity of a single layer Transformer layer is:*

$$O\left(B_w B_{W_c} L_\sigma B_{W_v}\left(\frac{2B_x^2\sqrt{C}}{\sqrt{m}}\left(1+\ln\left(\frac{\sqrt{m}}{2B_x\sqrt{C}}\right)\right)+B_x\sqrt{\frac{\ln(2d)}{m}}\right)\right)$$

The proof is left in Appendix Section A.3 for ease of presentation.

We can now take lemmas 2.3.3, 2.3.4, and 2.3.6 to get bounds on the Rademacher complexity. In the proof it can be seen the only matrix that needs to be covered in W_{QK}, thus we will only have a dependence on d and not k. We will show one corollary below and leave the rest to Appendix Section A.4.

Corollary 2.4.1.1. *Let us have the requirements needed for Theorem 2.4.1 along with* $\|W_{QK}\|_{1,1} \leq B_{W_{QK}}$ *Let*

$$B = B_w B_{W_c} L_\sigma B_{W_v}$$

and let

$$\alpha = B_{W_{QK}}\sqrt{2\log(2d^2+1)}$$

Then we have our Transformer Rademacher complexity being less than

$$O\left(B\left(\frac{B_x^3\alpha}{\sqrt{m}}\left(1+\ln\left(\frac{\sqrt{m}}{B_x^2\alpha}\right)\right)+B_x\sqrt{\frac{\ln(2d)}{m}}\right)\right)$$

2.4.2 Single Layer Multiple Heads

Let $H \in \mathbb{N}$ and let Y_j, $j \in [H]$ each be a Transformer head. Then notice by linearity of expectation:

$$\mathbb{E}\left[\sup_{Y_1,\ldots,Y_H}\sum_{i=1}^{m}\epsilon_i w^\top \sum_{j=1}^{H} Y_j(X_i)\right] = \sum_{j=1}^{H}\mathbb{E}\left[\sup_{Y_j}\sum_{i=1}^{m}\epsilon_i w^\top Y_j(X_i)\right]$$

The expectation above is the same as the single layer, thus multiple heads just adds a linear H term to the Rademacher complexity.

2.4.3 Multiple Layers

We have seen that we are able to get sequence length independent Rademacher complexities (and therefore generalization bounds) for a single layer Transformer architecture. For multiple layers, it suffices to take the proof found in Edelman et al. [2022] and slightly rework it so that it will work for an arbitrary linear covering number bound.

Theorem 2.4.2. *(Slight Reworking of Theorem A.17 in Edelman et al., 2022) Suppose we have a log covering numbers in the form of C_1/ϵ^2 and C_{B_x}/ϵ^2 for the function class $\{x \to Wx \mid x \in \mathcal{X}, w \in \mathcal{W}\}$ where $\|x\|_2 \leq 1 \ \forall x \in \mathcal{X}$ and $\|x\|_2 \leq B_x \ \forall x \in \mathcal{X}$ respectively. Suppose we also have $\left\|W_c^{(i)\top}\right\|_2 \leq B_{c2}$, $\left\|W_v^{(i)\top}\right\|_2 \leq B_{v2}$, $\left\|W_{QK}^{(i)}\right\|_2 \leq B_{QK2}$, $\|w\| \leq B_w$ along with them meeting the needed covering number restrictions. Let*

$$\begin{aligned}
\alpha_i &= \prod_{j=i+1}^{L} L_\sigma B_{c2} B_{v2}(1 + 4B_{QK2}) \\
\tau_i &= \alpha_i^{2/3} + (2\alpha_i L_\sigma B_{c2} B_{v2})^{2/3} + (\alpha_i L_\sigma B_{v2})^{2/3} \\
\gamma &= C_{B_x}^{1/3} (2L_\sigma B_{c2} B_{v2} \alpha_1 B_w)^{2/3} + C_1^{1/3} \left(1 + (B_w L_\sigma B_{v2})^{2/3}\right) \\
\eta &= C_1^{1/3} \left(B_w^{2/3} \sum_{i=2}^{L} \tau_i\right)
\end{aligned}$$

Then, the log covering number of g_{scalar}^{L+1} is

$$\frac{(\gamma + \eta)^3}{\epsilon^2}$$

The proof is left in Appendix Section A.5 for ease of presentation.

Notice this is the covering number of the entire multi-layer Transformer. Thus, we can recover an upper bound for the Rademacher complexity of it by using Dudley's integral.

Substituting our covering number bounds into theorem 2.4.2 gives us the three corollaries. We state one below and leave the rest to Appendix Section A.6.

Corollary 2.4.2.1. *Suppose we have the norm bounds required in lemma 2.3.6 for each $W_c^{(i)}, W_v^{(i)}, W_{QK}^{(i)}, w$ and let the maximum be B. Let B_x be the input bound. Suppose we also*

have the bounds needed for theorem 2.4.2. Let

$$\alpha_i = \prod_{j=i+1}^{L} L_\sigma B_{c2} B_{v2}(1 + 4B_{QK2})$$
$$\tau_i = \alpha_i^{2/3} + (2\alpha_i L_\sigma B_{c2} B_{v2})^{2/3} + (\alpha_i L_\sigma B_{v2})^{2/3}$$
$$\gamma = \left(B^2 B_x^2 \ln(2dk+1)\right)^{1/3} (2L_\sigma B_{c2} B_{v2} \alpha_1 B_w)^{2/3} +$$
$$\left(B^2 \ln(2dk+1)\right)^{1/3} \left(1 + (B_w L_\sigma B_{v2})^{2/3}\right)$$
$$\eta = \left(B^2 \ln(2dk+1)\right)^{1/3} \left(B_w^{2/3} \sum_{i=2}^{L} \tau_i\right)$$

Then, the log covering number of g_{scalar}^{L+1} *is*

$$\frac{(\gamma+\eta)^3}{\epsilon^2}$$

The above covering number is precise, but unwieldy to look at. To get a better sense of it, we can see that, ignoring polylog terms and constants, we get

$$B^2 B_x^2 B_w^2 (L_\sigma B_{c2} B_{v2} B_{QK2})^{O(L)} \frac{1}{\epsilon^2}$$

We do note that the multi-layer method does also work for one layer, however, it is not quite comparable to the bound found in Section 4.1 due to the norm restrictions being different. If we were to look at just the resulting values, the given single layer method essentially trades the cross product terms in the cubed factor for a factor of B_x^2, which seems like an acceptable trade. The proof for the single layer is also much more direct and easy to digest. It also gives a linear dependence on the amount of heads when the multi-layer method extended to multiple heads gives a dependence of $H^{1.5}$.

2.5 Theoretical Example: Word Prediction in NLP

Suppose we have a word embedding set up and a vocabulary of size K. One way to try to learn is by masking a certain percentage of words in a sentence and asking the Transformer to predict these words. Masking is done by taking the row that corresponds to the position of the masked word (let us call this row i) and giving as input a specific vector instead of the actual embedding of the word. Then the prediction is done by taking the vector in row i in the final layer of the Transformer and linearly transforming it into a size K vector. Then we can softmax this vector and use cross entropy loss to train. This is one of the ways BERT

[Devlin et al., 2018] is trained. Below, we will suppose only 1 word is masked for each input for ease of presentation. Let us use the cross entropy loss with softmax:

$$\ell_i(y, x) = -\sum_{i=1}^{k} y_i \log(\text{softmax}\,(x)_i)$$

where $y \in \{0,1\}^K$ is a one-hot encoded value that specifies the correct word and $x \in \mathbb{R}^K$ is the output of our Transformer at the masked index. Let us call this problem set up Transformer word masking. With this, we state the following theorem:

Theorem 2.5.1. *The Rademacher complexity of Transformer word masking can be found using Theorem 2.4.1 or Theorem 2.4.2, given the required norm assumptions and layer size for the theorems hold.*

Proof. First, we show this loss function is 2-Lipshitz in the ℓ_∞ norm. We prove this in Appendix A.7.

Therefore, if we let W be our linear map from the Transformer row to the vocabulary scores and let $Y_L^{(i)}$ be the output of our L-layer Transformer on the i^{th} sample. We then have by Foster and Rakhlin [2019]:

$$\begin{aligned}
&\mathbb{E}\left[\sup_{W,Y}\sum_{i=1}^{m}\epsilon_i\ell_i\left(W(Y_L^{(i)})_\tau\right)\right] \leq \\
&\tilde{\mathcal{O}}(\sqrt{K})\max_s \mathbb{E}\left[\sup_{W,Y}\sum_{i=1}^{m}\epsilon_i\left(W(Y_L^{(i)})_\tau\right)_s\right] = \\
&\tilde{\mathcal{O}}(\sqrt{K})\max_s \mathbb{E}\left[\sup_{W_s,Y}\sum_{i=1}^{m}\epsilon_i W_s(Y_L^{(i)})_\tau\right]
\end{aligned}$$

Notice that $W_s^\top \in \mathbb{R}^d$ and then by definition $(Y_L^{(i)})_\tau \in \mathbb{R}^d$ is a token from our Transformer output. Thus, the resulting function class in the Rademacher complexity term is of the same form needed to use Theorem 2.4.1 if $L = 1$ or Theorem 2.4.2 for $L \geq 1$. Therefore, we can use theorems to find the Rademacher complexity of Transformer word masking if the norm restrictions for the theorems hold. □

In order to get the generalization bounds from this Rademacher complexity, we require a bounded loss function. Using clipped cross entropy has been shown to have advantages over unbounded cross entropy [Hurtik et al., 2022, Wei et al., 2023], so using such a loss will allow us to get our generalization bounds from this set up.

2.6 Empirical example: Sparse Majority

The above sections show that, with bounded norms on the weight matrices, our generalization gap should not grow with sequence length. Thus, in this section, we will discuss a simulated study to see empirically if we find results that match our theory. We run a single layer Transformer on a simulated sparse majority data set on a variety of sequence lengths and we look at three results: (1) The total 1-norm of the weights in the Transformer, (2) The cross entropy generalization gap of the best epoch, (3) The validation accuracy of the best epoch for each sequence length.

The first two will show whether or not our theoretical findings are found in practice as well. The last one is more of a practical concern–a situation where the generalization gap is small but the network does not learn is not very useful.

The dataset we create is a sequence of zeros and ones where the label is determined by a majority of a sparse set of the indices. More concretely, if we have a sequence S of length is T, we have a set of indices is I, $|I| < T$, the label is

$$y_i = \mathbb{1}_{\{\sum_{i \in I} S_i > \frac{|I|}{2}\}}$$

In order to more accurately emulate real uses of Transformers, we embed 0 and 1 each into a d-dimensional vector where these two are orthogonal from each other. We also add the positional encoding defined by Vaswani et al. [2017] to add positional information to the sequence.

For our experiment we used a single layer of Tensorflow's `MultiHeadAttention` layer along with a second layer that extracts the $[CLS]$ layer and linearly transforms it into a vector of size 2. The loss we use is the cross entropy loss. Notice how we can use Section 2.5 to make this fit in the regime we have been discussing in this chapter; we can act as if the $[CLS]$ index is always masked and we have a vocabulary of size 2 (0 and 1).

The multihead attention layer has embedding dimension of 64 and 2 heads. The embedding dimension was chosen to be large while still allowing for moderate computation time. Only two heads were also chosen as well for computation time concerns. For our dataset, we had the sparse index set cardinality to 9 and used 300 training samples on sequence length 20, 40, 60, ..., 200 with a validation set size of 10000. The index set cardinality and training set size were chosen after finding a small enough size where the smaller sequence lengths could not always get perfect validation accuracy.

The Transformer trained on a NVIDIA Tesla V100 GPU for 200000 epochs with a batch size of 128. 200000 epochs, while a lot, was needed to allow for the larger sequence lengths to start to overfit. This batch size was also chosen after trial and error.

We trained our model for each sequence length 5 times (new data set each time) and recorded the 1-norms of the weights, the accuracy, and generalization gap of the best epoch. Figure 2.2 shows the worst generalization gap for each sequence length. Figure 2.1 shows the largest 1-norm of the weights for each sequence length. Figure 2.3 shows the best accuracy for each sequence length. We note that each of these do not necessarily represent the same run per sequence length.

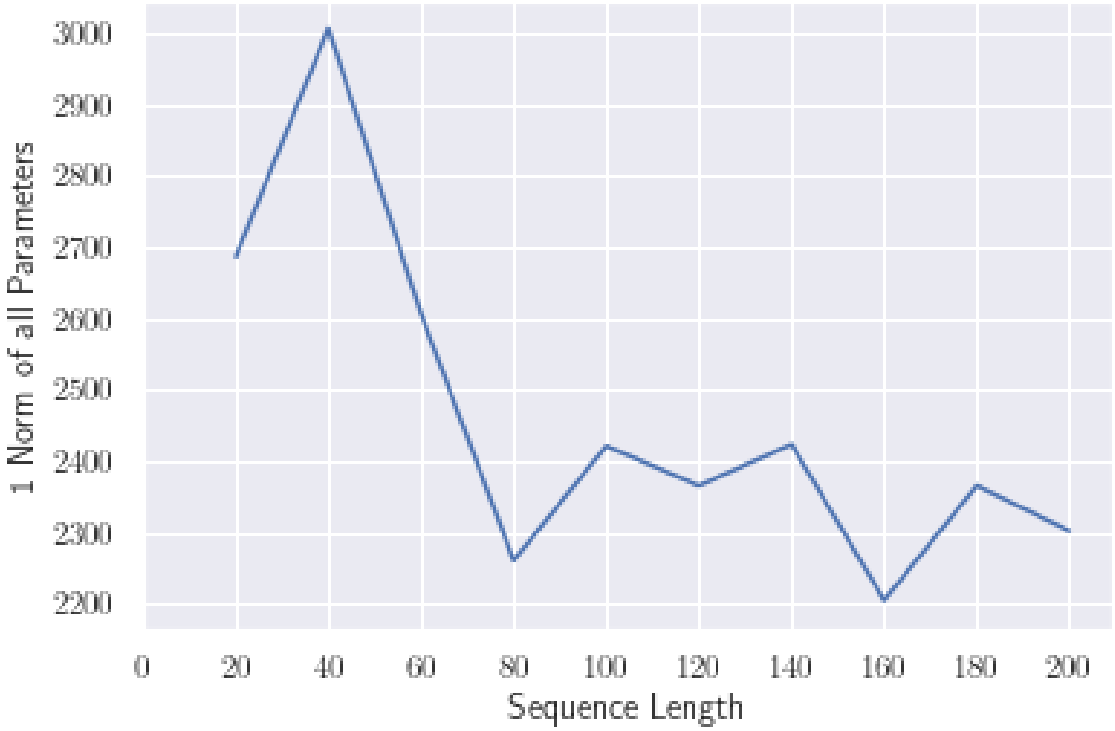


Figure 2.1: Plot of the max sum of the absolute value of all the weights across sequence lengths. The lack of any increasing trend further validates our assumption of bounded weights being credible.

As we can see, the figure 2.1 shows that the weights do not increase with sequence length, lending strength to our matrix norm assumptions.

We can also see in figure 2.2 the generalization gap also has no discernible trend and figure 2.3 shows the accuracy plateaus as sequence length increases. These results further help validate our theoretical findings and add evidence that longer sequence lengths do not inhibit how well Transformers learn.

The code for these experiments can be found here.

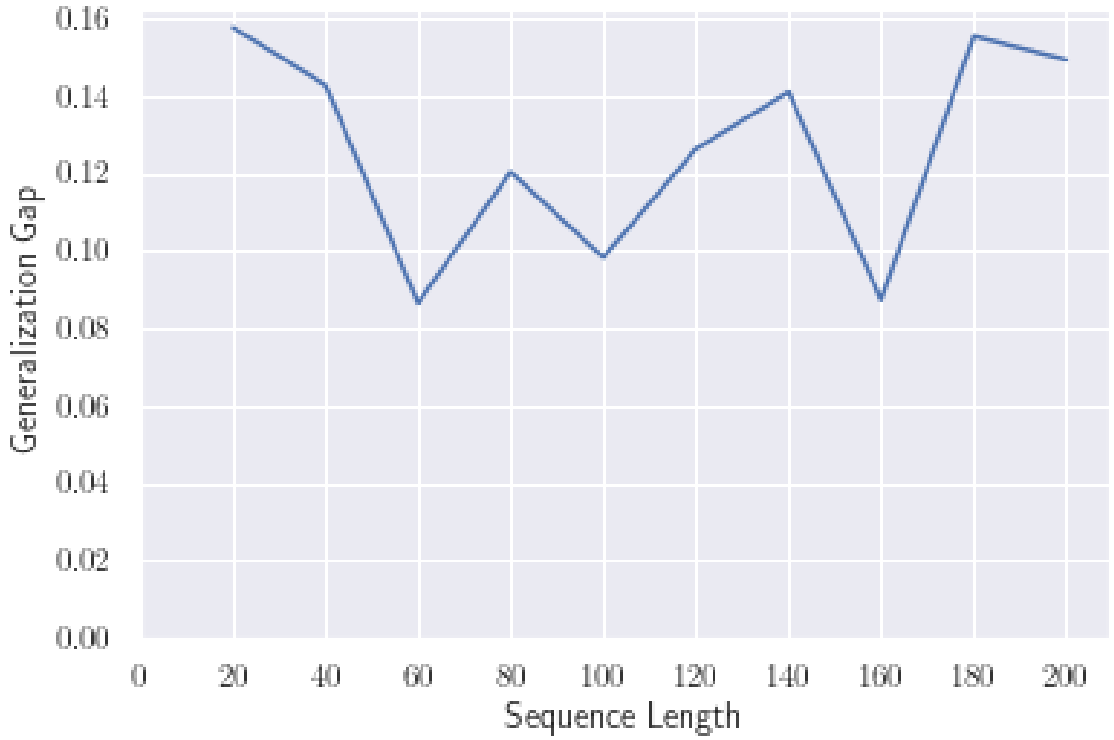


Figure 2.2: Plot of the max generalization gap across sequence lengths. There is no discernible trend in the plot, giving empirical validation to our theoretical results

2.7 Conclusion and Future Work

In this work, we give norm-based generalization bounds that do not grow with sequence length. This fills a hole in the literature where we can now have sequence length independent generalization bounds with the good properties the norm-based bounds give. We also give empirical evidence to validate our theoretical assumptions and theorems.

Future work could include sharpening the linear covering number bounds and generalizing them for more types of matrix/input norm bound combinations. Another avenue could be analyzing exactly how the norm-based bounds and parameter counting bounds trade off with each other.

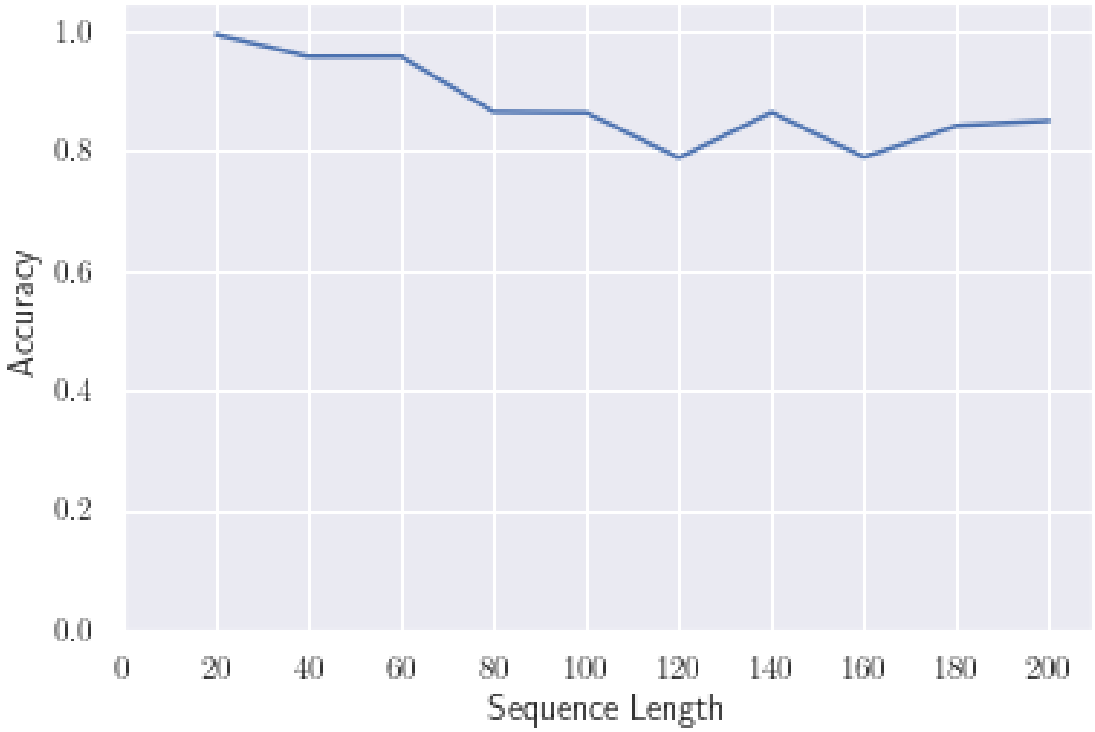


Figure 2.3: Plot of the maximum accuracy across sequence lengths. While this chapter makes no claims on the accuracy, we can see the graph does plateau. Therefore showing our models were learning well even at longer sequence lengths.

CHAPTER 3

On Next-Token Prediction in LLMs: How End Goals Determine the Consistency of Decoding Algorithms

Probabilistic next-token prediction trained using cross-entropy loss is the basis of most large language models. Given a sequence of previous values, next-token prediction assigns a probability to each possible next value in the vocabulary. From this, there are many ways to turn these next-token predictions into token sequences. This chapter examines a few of these algorithms (greedy/lookahead decoding, random sampling, and temperature-scaled random sampling) and studies their consistency with respect to the user end goals of information retrieval and creative generation through encoding these goals as loss functions. Although the consistency of surrogate losses with respect to a target loss function is a well researched topic, we are the first to study it in the context of LLMs (to the best of our knowledge). We find that, so long as next-token prediction converges to its true probability distribution, random sampling is consistent with outputting sequences that mimic sampling from the true probability distribution. For the other goals, such as minimizing the 0-1 loss on the entire sequence, we show that deterministic decoders have the edge over stochastic decoders. From these results, we see that there is a dichotomy created between the goals of information retrieval and creative generation for the decoding algorithms. This shows that choosing the correct decoding algorithm based on the desired goal is extremely important and many of the ones used are lacking theoretical grounding in numerous scenarios. While there has been evidence for this empirically, this chapter gives rigorous theoretical grounding to these results.

3.1 Introduction

The basis for nearly all large language models today is next-token prediction trained by minimizing the cross entropy loss function [Radford et al., 2018, Brown et al., 2020, Chowdhery et al., 2023, Touvron et al., 2023, Devlin et al., 2019]. However, next-token prediction only gives probabilities for the next token. Many tasks, such as machine translation or text gener-

ation, require an output of a token sequence. Thus, we must have some decoding algorithm that takes these next-token predictions and outputs a sequence. With a large amount of test-time computation being used in state-of-the-art models [Jaech et al., 2024, Guo et al., 2025], the mathematical foundations of these algorithms are of great interest.

In this chapter, we analyze the behavior of next-token prediction and how the choice of decoding algorithms can impact the asymptotic optimality of the outputs. This work can be thought of as studying surrogate loss consistency: we minimize a surrogate loss function (cross entropy on next-token prediction), but are interested in a different target loss function (e.g., Hamming loss between predicted and correct sequence). This notion of consistency has been extensively studied in machine learning. Bartlett et al. [2006] showed results for binary classification where they minimize a surrogate loss function and show consistency with respect to the 0-1 loss. Tewari and Bartlett [2007] extend this approach to multi-class classification. Gao and Zhou [2011], Koyejo et al. [2015], Wu and Zhu [2020] all have worked on consistency for multi-label classification.

There has also been research into next-token prediction and decoding algorithms. Saunshi et al. [2021] investigates how linearly transforming next word prediction can predict text classification. Li et al. [2024] studies what can be learned by a single attention layer for next-token prediction. Snell et al. [2024], Wiher et al. [2022], Shi et al. [2024] all investigate a few types of decoding algorithms and empirically evaluate them. There has also been much research on how next-token prediction learns [Bachmann and Nagarajan, 2024, Lin et al., 2025, Thrampoulidis, 2024], but, as far as we are aware, we are the first to investigate the consistency of next-token prediction and these decoding algorithms.

It is standard to analyze consistency in an asymptotic setting of sufficiently large sample sizes and models where the surrogate loss has been fully minimized. We therefore assume that our next-token predictor converges to the true next-token distribution and we only have query access to it. Note that this emulates the training of our next-token predictor as asymptotic minimization of cross-entropy results in correct next-token distributions. Given this, we investigate our decoding algorithms with regards to two high-level goals central to how large language models are used today: information retrieval, where the user is looking for a "correct answer" and creative generation, where the user is looking for new samples from the distribution of human language [Paaß and Giesselbach, 2023, Petroni et al., 2019, Brown et al., 2020]. We do this by minimizing a loss function that acts as a proxy for these goals: the N-gram Hamming loss for correct information retrieval and the cross entropy loss for the entire sequence for generating samples.

This chapter gives a framework to study the consistency of various decoding algorithms with respect to different high level goals. We show that only deterministic decoding algo-

rithms can be consistent for the N-gram Hamming loss, however, these have infinite loss for the cross entropy loss for any non-deterministic true sequence output distribution. Therefore, stochastic decoders are necessary to have a non-infinite loss for the cross entropy objective, but fail at consistency for the N-gram Hamming loss. This shows there is no one-size-fits-all decoding algorithm and one must adapt their decoder to their desired goal. While this dichotomy has been empirically studied [Wiher et al., 2022, Shi et al., 2024], we are the first to give theoretical justification to this phenomenon as far as we are aware. There has also been a recent line of work on adaptive decoding strategies [Zhu et al., 2024b,a, Dhuliawala et al., 2024]. These decoding strategies change the decoder output distribution to be more or less determinsitic-like depending on some criteria which aligns itself with the information retrieval versus creative generation dichotomy. Therefore, our theoretical results are consistent with these recent empirical findings.

We also show that there is no consistent polynomial-time decoding algorithm for all output distributions for the N-gram Hamming loss and we create a small Markov chain experiment to see the optimality of deterministic decoders. For the cross entropy loss, we show random sampling is consistent for all probability distributions over the sequence outputs. Finally, we give a rate for the suboptimality gap of the expected risk for temperature scaling with respect to the temperature parameter.

This chapter is organized as follows: Section 4.2 goes over the requisite background and notation needed for this chapter, including the decoding algorithms studied in this chapter. Section 4.3 discusses the problem set up. Section 3.4 goes over the case where the goal is information retrieval. Finally, section 3.5 shows results for when the goal is creative generation.

3.2 Notation and Background

For notation, we will use $\mathcal{Y}$ as an output space, $\mathcal{X}$ as an input space, and ℓ as a loss function. Since the outputs are sequences, $y \in \mathcal{Y}$ will refer to the entire sequence, y_i will refer to index i in the sequence, and $y_{[i]}$ will refer to the subsequence from indices 1 to i. In general, $[j]$ is the ordered set $(1, 2, \ldots, j)$ and $y_{i:j} = y_i y_{i+1} \ldots y_j$. For two strings, the $+$ operator will mean concatenation. For probabilities, when one sees $p(v \mid y_{[i-1]})$, this is the conditional probability of the v at index i given the sequence $y_{[i-1]}$. We will often write this as $p(y_i \mid y_{[i-1]})$. To keep consistent with this notation, we will write $p(y_i)$ as the marginal of the distribution at index i for the token y.

3.2.1 Expected Risk

Given a loss function ℓ, a probability distribution p over inputs $\mathcal{X}$ and outputs $\mathcal{Y}$, and a hypothesis $h : \mathcal{X} \to \mathcal{Y}$, the expected risk is normally defined as follows:

$$R(h, p, \ell) = \mathop{\mathbb{E}}_{(x,y)\sim p} \left[\ell(h(x), y)\right].$$

This value is oftentimes what is trying to be minimized when training a machine learning algorithm [Shalev-Shwartz and Ben-David, 2014b, Bartlett et al., 2006, Tewari and Bartlett, 2007, Gao and Zhou, 2011], however, this set up does not have the required granularity needed for our purposes. There has been work that has modified the expected risk so that it can fit their use cases, such as needing a "pred" function to convert the output of their algorithm into a prediction [Tewari and Bartlett, 2007, Ramaswamy et al., 2013, Ramaswamy and Agarwal, 2016]. We will then also reformulate the expected risk so that it fits to our problem.

We assume we have access to a next-token predictor, which can only output conditional probabilities for the next token given the previous tokens and input. The notation for this will be $p_{ntp}(y_i \mid y_{[i-1]})$ for every y_i in our vocabulary. These conditional distributions naturally induce a unique probability distribution on the entire sequence, thus we will interchangeably refer to p_{ntp} as probability distribution itself. A *ntp* subscript on a probability distribution is used to emphasize that the distribution is being used as a next-token predictor. Given p_{ntp}, we require a decoding algorithm $\mathcal{D}$, which takes in an input $x \in \mathcal{X}$ and a next-token predictor p_{ntp} ($\mathcal{D}$ only has access to conditional distributions of p_{ntp}) and uses them to output a sequence $\hat{y} \in \mathcal{Y}$. Since $\mathcal{D}$ can have internal randomness, we will refer to the distribution of outputs produced by $\mathcal{D}$ as $p_{\mathcal{D}(p_{ntp})|x}$. We will then define our expected risk as follows:

$$R(\mathcal{D}, p, p_{ntp}, \ell) = \mathop{\mathbb{E}}_{x\sim p_x} \left[\mathop{\mathbb{E}}_{y\sim p|x, \hat{y}\sim p_{\mathcal{D}(p_{ntp})|x}} \left[\ell(\hat{y}, y)\right] \right],$$

where $p \mid x$ is the conditional distribution of the output y given an input x.

3.2.2 Decoders

We will look at 3 types of decoding algorithms: K_T-lookahead decoding, random sampling, and temperature-scaled random sampling.

3.2.2.1 K_T-Lookahead

The word "lookahead" has been used in a few different ways in LLM decoding [Snell et al., 2024, Fu et al., 2024]. Here, we will define the K_T-lookahead algorithm as a generalization of the well-known "greedy" decoding algorithm [Shi et al., 2024, Wiher et al., 2022]. For choosing the next token(s), we will find all K-length combinations of our tokens and then keep the first T tokens of the maximum K-length sequence.

Algorithm 1 K_T-lookahead

Require: $L \in \mathbb{N}$, $K \leq L$, $T \leq K$, $p_{ntp}(\cdot \mid \cdot)$, Vocabulary $\mathcal{V}$

$y \leftarrow$ ""

while length$(y) < L$ **do**

$c \leftarrow \max\{K, L - \text{length}(y)\}$

$y' = \arg\max_{v_1,\ldots,v_c \in \mathcal{V}} \{p_{ntp}(v_1 \mid y) p_{ntp}(v_2 \mid y + v_1) \ldots p_{ntp}(v_c \mid y + v_{1:c-1})\}$

$H \leftarrow \min\{T, L - \text{length}(y)\}$

$y \leftarrow y + y'_{[H]}$

end while

We note that $K = T = 1$ is greedy decoding.

3.2.2.2 Random Sampling

Since next-token prediction outputs probabilities, random sampling will choose the next token given these conditional probabilities.

Algorithm 2 Random Sampling

Require: $L \in \mathbb{N}$, $p_{ntp}(\cdot \mid \cdot)$, Vocabulary $\mathcal{V}$

$y \leftarrow$ ""

while length$(y) < L$ **do**

$y_{new} \sim p_{ntp}(\cdot \mid y)$

$y \leftarrow y + y_{new}$

end while

3.2.2.3 Temperature-Scaled Random Sampling

Temperature scaling is where one scales the next-token probabilities to encourage or discourage exploration of the space. It is used in almost all, if not all, large language models [Brown et al., 2020, Achiam et al., 2023, Chowdhery et al., 2023, Touvron et al., 2023].

Normally the probabilities are found by using a softmax on logits z_i. Temperature scaling is then done by using softmax on z_i/T, where T is the temperature parameter. However,

using only the probabilities themselves, we can do temperature scaling using temperature γ as:

$$p_{scaled}(y_i \mid y_{[i-1]}, x) = \frac{p(y_i \mid y, x)^\gamma}{\sum_{v \in \mathcal{V}} p(v \mid y, x)^\gamma}.$$

We show the equivalence in Appendix B.2.1.

For this decoding algorithm, we will be randomly sampling the next token from the temperature-scaled distribution.

Algorithm 3 Temperature Scaled Random Sampling

Require: $L \in \mathbb{N}$, $\gamma > 0$, $p_{ntp}(\cdot \mid \cdot)$, Vocabulary $\mathcal{V}$

$y \leftarrow$ ""

while length$(y) < L$ **do**

$p_\gamma(v \mid y) = \frac{p_{ntp}(v|y)^\gamma}{\sum_{u \in \mathcal{V}} p_{ntp}(u|y)^\gamma}$

$y_{new} \sim p_\gamma(\cdot \mid y)$

$y \leftarrow y + y_{new}$

end while

3.3 Problem Setup

Let $\mathcal{X}$ be a general input space. Let $\mathcal{V}$ be a vocabulary, $*$ be a null character, and let $L \in \mathbb{N}$. Then, let $\mathcal{Y} \subseteq \{y_1 y_2 \dots y_j \underbrace{***}_{L-j \text{ indices}} \mid y_i \in \mathcal{V}, j \leq L\}$ be our sequence output space. Each $y \in \mathcal{Y}$ is thus a sequence padded to a finite maximum length using the null character. We will also assume for any next-token predictor, if the current string has $*$ in it, all mass for the next token is at $*$. This is done as $*$ represents empty space and is only used in the analysis to simplify dealing with strings of different lengths. The Transformer architecture has a maximum sequence length it can output, thus this set up does not lose any generality to modern day large language models.

Let us represent the true probability distribution as p^* over $\mathcal{X} \times \mathcal{Y}$. Given an initial next-token predictor p^0_{ntp}, we will assume that it is iteratively updated using cross entropy on the next-token distributions. Let each new iteration be p^i_{ntp}.

Assumption 3.3.1. $\forall y \in \mathcal{Y}$, $\forall i \in [L] \quad p^i_{ntp}(y_i \mid y_{[i-1]}) \to p^*(y_i \mid y_{[i-1]})$ *in KL-Divergence.*

It is standard to study surrogate loss consistency when the surrogate loss is asymptotically minimized [Gao and Zhou, 2011, Tewari and Bartlett, 2007]. It can be easily seen that minimizing cross entropy implies the KL-Divergence is 0. From a practical standpoint,

this assumption is credible as, given proper data, modelling, and updating, the next-token conditional distributions will converge to the true conditional distribution through the minimization of cross entropy. We then show in Appendix B.2.2 that this also implies $p_{ntp} \to p^*$ in KL-Divergence as well.

Now, given $p^i_{ntp} \to p^*$, this chapter studies when our decoding algorithms have the property

$$R(\mathcal{D}, p^*, p^i_{ntp}, \ell) \to \inf_{h:\mathcal{X}\to\mathcal{Y}} R(h, p^*, \ell).$$

3.4 Consistency for N-Gram Hamming Loss

Historically, N-grams have been important in sequence metrics like the BLEU score [Papineni et al., 2002] and the ROUGE-N score [Lin, 2004]. N-grams are used to segment a sequence into portions evaluate the correctness of each portion. We will take this idea and define a new loss function, which we call the N-gram Hamming loss. Mathematically, we define it as:

$$\sum_{i=1}^{L-N+1} 1_{\{\hat{y}_{i:i+N-1} \neq y_{i:i+N-1}\}}.$$

For $N = 1$ this is the Hamming loss and for when $N = L$ we have the 0-1 loss, which themselves are two canonical loss functions in machine learning. Intermediate losses when $N \in [2, L-1]$ might be useful in their own right, but here we consider them as a mathematically tractable representative for the various N-gram based metrics used in sequence learning.

We want to determine for which probability distributions will our decoding algorithms always produce the optimal output for all sets of inputs with positive measure. Below we show what is optimal for the N-gram Hamming loss:

Lemma 3.4.1. *Let p be a probability distribution over output sequences and let*

$$g(y) = \sum_{i=1}^{L-N+1} p(y_{i:i+N-1}).$$

Then, the optimal output for N-gram Hamming is

$$\arg\max_y \{g(y)\}.$$

The proof of is left to the Appendix B.2.4 for ease of presentation. Note how this generalizes the already known optimal outputs for the Hamming and 0-1 loss [Dembczyński et al., 2010].

3.4.1 Exchanging Consistency for Optimality

Here we give a useful result that will allow us exchange the limit and expectation in the expected risk, given a decoder meets the assumptions needed.

Proposition 1. *Suppose $p_{\mathcal{D}(p_{ntp})|x}$ is the probability distribution of the output of $\mathcal{D}(p_{ntp}) \mid x$. Then, given an M-bounded loss function ℓ and*

$$\forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \lim_{i \to \infty} \left(p_{\mathcal{D}(p^i_{ntp})|x}(y) - p_{\mathcal{D}(p^*_{ntp})|x}(y) \right) = 0.$$

Then

$$\lim_{i \to \infty} R(\mathcal{D}, p^*, p^i_{ntp}, \ell) = R(\mathcal{D}, p^*, p^*_{ntp}, \ell).$$

We leave the proof to Appendix B.2.3 for ease of presentation. This allows us to deal with the optimality of our decoding algorithms given a true sequence distribution p^* instead of the consistency of a sequence p^i. The assumption is also very reasonable, if not even a desirable trait for a decoder to have: it says that as p^i converges to p^*, the probability of our decoder outputting any sequence using p^i_{ntp} should converge to the probability of our decoder outputting that sequence under p^*_{ntp}.

3.4.2 Optimal Decoding for All Probability Distributions is Not in Polynomial Time

To motivate the usage of various decoding algorithms for the N-gram Hamming loss, we will show that, even if we have access to next-token predictons from the true sequence distribution (p^*), there does not exist a polynomial-time (in sequence length L) decoding algorithm that is optimal for all probability distributions. We show in Section 3.4.4 that stochastic decoders (i.e., decoders that can sometimes choose one value or another depending on internal randomness) can be not optimal so long as they put non-zero mass on an non-optimal output. Thus we are left with two types of decoders: deterministic decoders and stochastic decoders that put all the probability mass on optimal outputs. Let us call the latter optimal stochastic decoders.

Theorem 3.4.2. *Let $\mathcal{V}$ be a vocabulary and let $\mathcal{Y}$ have maximum length L. Let p be such that*

$$\forall y \in \mathcal{Y}, \ \forall i \in L \quad p(y_i | y_{[i-1]}) = \frac{1}{|\mathcal{V}|}.$$

Then, any optimal deterministic or optimal stochastic decoder algorithm $\mathcal{D}$ for the N-gram Hamming loss must have a runtime of at least $C(|\mathcal{V}|^L - 1)$, assuming queries to the next-token predictor take C time.

The proof is left to Appendix B.2.5. The idea of the proof relies on the pigeonhole principle. If there are an exponential amount of values $p(v|y_{[j-1]})$ that can be queried, if we only query a polynomial amount of them, there are many adversarial probability distributions that fit the queried values, but have different optimal values. Even though the proof is done on the uniform distribution, one can see how this idea can fit a large class of sequence probability distributions.

Corollary 3.4.2.1. *Optimal decoding of the N-gram Hamming problem takes exponential time in L assuming black-box access to next-token probabilities.*

Proof. Since accessing from memory is assumed to be constant time, we have shown there is a distribution that will take $\Omega(|\mathcal{V}|^L)$ runtime. This, combined with the results in Section 3.4.4, gives us our result. □

In modern large language models, these next-token probabilities are done using calls to a Transformer architecture, which do not have $O(1)$ runtime. However, for the N-gram Hamming loss, we see that next-token prediction has an exponential lower bound for optimality that no amount of Transformer optimization can fix.

3.4.3 K_T-Lookahead Decoding

For this subsection, ties are a curse. When choosing the next token(s), if we have at least 2 sequences in our next token(s) $\arg\max$ such that at least 1 puts the algorithm not on a path to an overall sequence $\arg\max$, then there is no way for our algorithm to be optimal for all probability distributions. This is because there are two ways to break ties: deterministic breaks or random breaks. For a deterministic tiebreak, we can adversarially create a probability distribution where we choose wrong. For random tiebreaks, we show in in Section 3.4.4 that it also can not be optimal. Therefore, we will only look at the class of probability distributions where we will not run into any ties in any of our $\arg\max$s. Let this class be called $\mathcal{P}$.

We note that restricting to this set means that we can not use the example given in the proof of optimal decoding requiring exponential time. However, the proof does not rely on ties, it instead relies on having no few conditional probabilities $p(v|y_{[i-1]})$ being large enough to make all the others at that level irrelevant. Thus, one can imagine extremely small perturbations to the example's conditional distributions such that the distribution is

in $\mathcal{P}$, but is still very close to the uniform distribution. The rest of the proof would then work out the same.

Lemma 3.4.3. *K_T-lookahead decoding meets the criteria to use Proposition 1*

The proof is left to Appendix B.2.6 for ease of presentation. Using this lemma, we only need to concern ourselves with p^* when trying to show consistency.

Now, in order for K_T-lookahead decoding to be optimal for the N-gram Hamming loss, we give a reframing of K_T-lookahead in terms of what probability distributions it is optimal for below:

Proposition 2. *Let us have a probability distribution $p^* \in \mathcal{P}$ over $\mathcal{X} \times \mathcal{Y}$ and let*

$$C = \{x \mid \arg\max_{y} \sum_{i=1}^{L-N+1} p^*(y_{i:i+N-1} \mid x) = y^\dagger \text{ where}$$
$$y^\dagger_{(Tc+1):\min\{(Tc+T),L\}} =$$
$$\left(\arg\max_{y_{(Tc+1):\min\{(Tc+K),L\}}} \{p^*(y_{(Tc+1):\min\{(Tc+K),L\}} \mid x, y^\dagger_{[Tc]})\} \right)_{[T]}$$
$$\text{for } c \in \mathbb{Z}_+, Tc \leq L - T\}.$$

Then, K_T-lookahead is N-gram Hamming loss optimal for p^* iff

$$p^*_x(C) = 1.$$

The proof is left to Appendix B.2.7.

Since K_T-lookahead is a greedy algorithm, this characterization can be interpreted as it only being optimal for probability distributions that lend themselves well to greedy algorithms. Thus, no matter how perfect the next-token predictor is, we are simply running a greedy algorithm and these algorithms will fall into the same traps greedy algorithms have been known to fall into. For example, in Appendix B.2.8 we give a Markov chain that is not optimal by exploiting the characterization above.

Given this, it is natural to then ask how often do these decoders run into such a problem. We set up a simulation study on Markov chains to empirically test optimality. We create each graph by having its starting distribution and each transition distribution be Dirichlet distributed with the parameters all being the same value, α. We group each graph by the α parameter used and then take the average amount of times K_T-lookahead was optimal. Each group has 200 graphs. We do note, while improbable, there can be ties in the K_T-lookahead arg max in our simulations and the ties are broken by which path was seen first. More details on the simulation study can be found in Appendix B.1.1.

While this is a simple experiment, we note that our results show that we can not do much more. Since we are looking for optmality and there is no polynomial-time optimal decoder, we must use an exponential-time algorithm to find the optimal sequence. This immediately narrows the experiments we can run since we can only use a model with a restrictively small vocabulary and maximum sequence length.

In Figure 3.1, we plot the percent of times K_1-lookahead was optimal in a group for the 1-gram Hamming loss against the average KL-Divergence from the uniform distribution for that group. We can see that it does not do well no matter how short the sequence is. Even for short sequences with low entropy distributions (very "peaky" distributions), K_1-lookahead decoding still guesses wrong about 10% of the time.

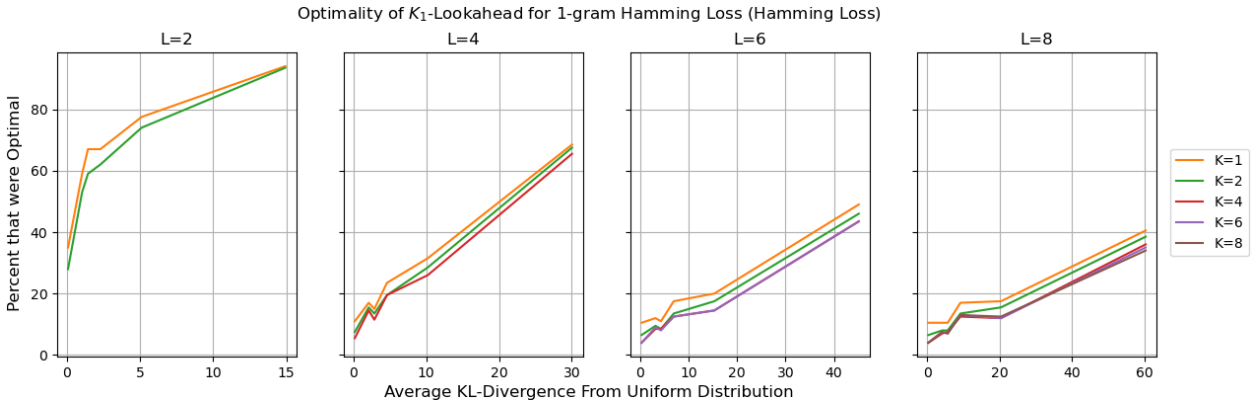


Figure 3.1: A plot of the amount of trials K_1-lookahead was optimal for the 1-gram Hamming loss (the Hamming loss). Each point represents the average optimality over 200 randomly generated Markov chains with a set amount of nodes and Dirichlet parameter α. Smaller αs create more "peaky" distributions and thus have higher KL-divergence from the uniform distribution, while larger αs create more uniform distributions. There were 8 nodes in each Markov chain for this figure and the sequence length goes up by two as one moves right in the plots.

The next figure, Figure 3.2, we see how K_1-lookahead does for the L-gram Hamming loss. When $K = L$, K_1-lookahead will be optimal, which is why each of them has one line that is perfect. For the rest of the lines, we see they do better than they did for the Hamming loss, which is interesting in its own right. This trend is generally seen when looking at the other values of N, L, K as well. We leave a more thorough explanation and analysis of our simulations to Appendix B.1.1, where we also empirically analyze K_K-lookahead decoding as well.

A beam search is, instead of greedily choosing at each step in the lookahead algorithm, one keeps the top B sequences at each step and uses them for the next steps. Once at the end, then the beam search chooses the best out of the B outputs [Shi et al., 2024, Wiher

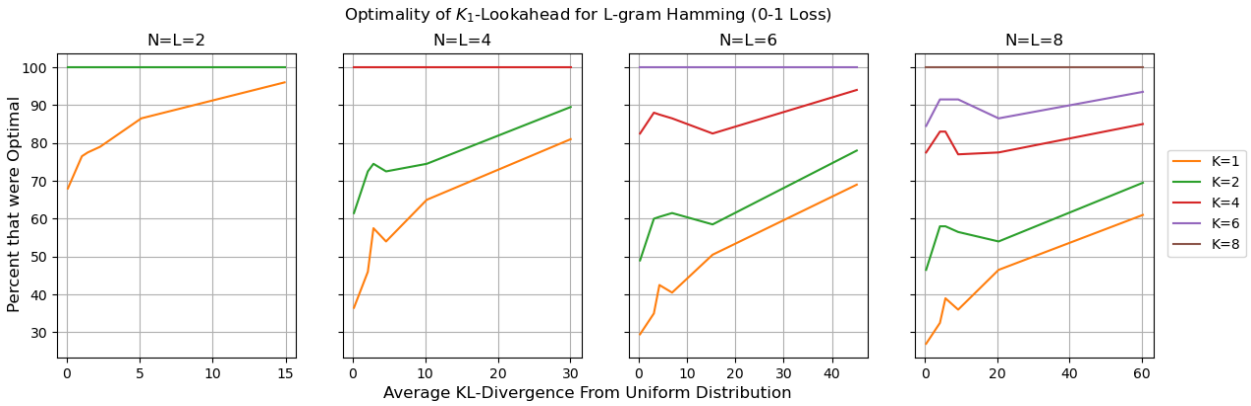


Figure 3.2: A plot of the amount of trials K_1-lookahead was optimal for the L-gram Hamming loss (the $0-1$ loss). The same setup as Figure 3.1 otherwise.

et al., 2022]. We can easily extend our reframing of K_T-lookahead to a K_T-lookahead beam search. Let $\arg\max \mathrm{B}$ be the set of the top B values.

Corollary 3.4.3.1. *Let us have our decoder be a K_T-lookahead beam search with beam width B. Let us have a probability distribution $p^* \in \mathcal{P}$ over $\mathcal{X} \times \mathcal{Y}$ and let*

$$C = \{x \mid \arg\max_y \sum_{i=1}^{L-N+1} p^*(y_{i:i+N-1} \mid x) = y^\dagger \text{ where}$$
$$y^\dagger_{(Tc+1):\min\{(Tc+T),L\}} \in$$
$$\left(\underset{y_{(Tc+1):\min\{(Tc+K),L\}}}{\arg\max \mathrm{B}} \{p^*(y_{(Tc+1):\min\{(Tc+K),L\}} \mid x, y^\dagger_{[Tc]})\} \right)_{[T]}$$
$$\text{for } c \in \mathbb{Z}_+, Tc \leq L - T\}.$$

Then, K_T-lookahead is N-Hamming loss optimal for p^* iff

$$p^*_x(C) = 1.$$

Proof. The same as Proposition 2, but with $\arg\max \mathrm{B}$ instead of $\arg\max$. □

Going back to greedy K_T-lookahead, let $\mathcal{P}_{K,T,N}$ be all probability distributions where K_T-lookahead is optimal with respect to N-gram Hamming.

One could expect that increasing K would monotonically increase $\mathcal{P}_{K,T,N}$ since the decoder is "looking" farther into the future. One would also expect that decreasing T would monotonically increase $\mathcal{P}_{K,T,N}$ since taking less tokens now will allow us to reconsider later when we have more information. However, we show below that neither of these are generally the case.

Proposition 3. *Let $K_1, K_2 \in [L-1]$, where $K_1 < K_2$. Then $\forall T_1 \in [K_1]$, $\forall T_2 \in [K_2]$, and $\forall N \in [L]$, we have $\mathcal{P}_{K_1,T_1,N} \not\subset \mathcal{P}_{K_2,,T_2N}$.*

Proposition 4. *Let $K \in \{2, 3 \ldots, L-1\}$, $N \in [L]$ and $T \in [K-1]$. Then, if $K < L - T$, we have $\mathcal{P}_{K,T+1,N} \not\subset \mathcal{P}_{K,T,N}$*

These proofs are left to Appendix B.2.9 and B.2.10, respectively. For Proposition 4, when $K \geq L - T$, there can exist monotonicity. For example, in Appendix B.2.10 we show it for when $N = L$.

These propositions show that the K and T hyperparameters have no general monotonicity and their optimality is distribution specific. In the case of large language models, since conditioning on different prompts changes the distribution, this shows that the optimal hyperparameters of K_T-lookahead are input dependent.

3.4.4 Stochastic Decoders

Below we show any decoder that is stochastic in any way is not optimal, so long as there is a non-zero probability of choosing a sequence that is not optimal. The proof is left to Appendix B.2.11.

Proposition 5. *Let our loss function be the N-gram Hamming loss. Let p be a probability distribution over $\mathcal{X} \times \mathcal{Y}$. Then, any stochastic decoder that has a non-zero probability of outputting a $\hat{y} \in \mathcal{Y}$ where $\hat{y} \notin \arg\max_y g(y)$ for a set of inputs that have non-zero probability is not optimal.*

3.4.4.1 Random Sampling and Temperature-Scaled Random Sampling

We show in Appendix B.2.12 that both of these decoding algorithms meet the criteria for Proposition 1. Thus, since each of the runtimes of these scale linearly with L, by Proposition 1 and Theorem 3.4.2, we have that neither of these decoders are consistent for all probability distributions. In fact, so long as $\gamma \neq \infty$, neither of these are consistent for any non-uniform or non-deterministic probability distribution in $\mathcal{P}$ by Proposition 5.

3.5 Consistency for Sample Generation

One valid goal of a large language model is to sample responses from the distribution of human speech [Paaß and Giesselbach, 2023]. We know that for any two probability distributions, minimizing cross entropy implies they will be equal to each other. Therefore, if we want our goal to be sampling from a true sequence probability distribution, the cross

entropy loss on the entire sequence is a natural choice. In this section we will not only use cross entropy to train our next-token predictor, but also use it as a loss function for our sequential output.

Given a decoding algorithm $\mathcal{D}$ and input $x \in \mathcal{X}$, the cross entropy loss is defined as follows:

$$CE(p^* \mid x,\ p_{\mathcal{D}(p_{ntp})|x}) = \mathop{\mathbb{E}}_{y \sim p^*|x} \left[-\log \left(p_{\mathcal{D}(p_{ntp})|x}(y) \right) \right].$$

3.5.1 Deterministic Decoders

For deterministic decoding algorithms, it is easy to see by the definition of cross entropy that for any non-deterministic probability distribution, these decoding algorithms will have infinite cross entropy. Thus, they are not consistent.

3.5.2 Random Sampling

In the N-gram Hamming loss setting, we saw that random sampling is not consistent for all non-deterministic or non-uniform probability distributions. Here, however, we will show it is consistent for every probability distribution. The proof is left to Appendix B.2.13.

Proposition 6. *Random sampling is always consistent under the cross entropy loss function in our setting.*

We note that, once we set up the surrogate loss framework, this essentially follows from the definition of autoregressive modeling. We believe this shows how natural this framework fits within this setting.

3.5.3 Temperature-Scaled Random Sampling

From the previous section we can see that when $\gamma = 1$, we know temperature-scaled random sampling is consistent with any true probability distribution. However, below we show it is not consistent for nearly all true probability distributions when $\gamma \neq 1$.

Proposition 7. *When $\gamma \neq 1$, temperature-scaled random sampling is only optimal when each next-token distribution is a uniform distribution or a deterministic distribution.*

The proof is left to Appendix B.2.14.

We next show how the expected risk increases as γ changes. For convenience, we assume below that for $\mathcal{Y} = \mathcal{V}^L$. The proof method works in generality, however, the resulting $\log(|V|)$ term in the lower bound would be much less interpretable. A more in-depth explanation is given at the end of Appendix B.2.15.

Proposition 8. *Let p be a probability distribution over $\mathcal{Y}$ and let p^γ be our temperature-scaled random sampled distribution with respect to p. Let opt be the minimum expected cross entropy obtainable with respect to p. Then, there exists constants $C_1, C_2, C_3 \in \mathbb{Z}_+$ that depend only on p such that we have:*

For $\gamma > 1$:

$$\gamma C_1 - opt \leq \mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(p^\gamma(y)\right)\right] - opt \leq$$
$$\gamma C_3 + L\log\left(|V|\right) - opt.$$

For $\gamma < 1$:

$$\left(L\log\left(|V|\right) - opt\right) - \gamma C_2 \leq \mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(p^\gamma(y)\right)\right] - opt \leq$$
$$\left(L\log\left(|V|\right) - opt\right) + \gamma C_3.$$

We leave the proof to Appendix B.2.15.

We require two bounds due to how the behavior changes from when $\gamma < 1$ to $\gamma > 1$. We know that as γ approaches ∞, our distribution becomes closer and closer to a point mass, thus our cross entropy goes to infinity. The inequalities set forth in when $\gamma > 1$ shows our loss goes to infinity at a rate of γ.

As γ approaches 0, we know our distribution will get closer and closer to the uniform distribution. Thus, our cross entropy should go to $-\log\left(|V|^{-L}\right) = L\log\left(|V|\right)$, which we can see through the inequalities when $\gamma < 1$ that it also goes to $L\log\left(|V|\right)$ as well at a rate of γ.

From this, we see temperature scaling behaves asymptotically as is expected in its scaling parameter and it does so linearly.

3.5.4 Proper Losses

A proper loss function with regards to this work is a loss function whose minimum is obtained by the true distribution. One well-used proper loss is the cross entropy loss and others include the Brier loss and the spherical loss [Vovk, 2015].

Corollary 3.5.0.1. *All the results regarding cross entropy loss from Sections 3.5.1 and 3.5.2 hold for any proper loss.*

Proof. Cross entropy is a proper loss and since random sampling decoding is consistent with it, we have that random sampling is consistent with respect to any proper loss. Further, since we see that no deterministic function is consistent for non-deterministic probability

distributions, we have that they do not limit to the true distribution. Thus, they can not be consistent for other proper losses as well. □

This also shows how useful the surrogate loss consistency framework can be. Due to how autoregressive modeling was created, the proof for Proposition 6 follows very easily once we have set up our framework. If one wanted to directly show the same result for other proper losses, it would most likely be much harder. With this framework we are able to show the result for the one loss function where it is easy and have it apply to all other proper losses.

3.6 Conclusion and Future Work

In this chapter we explored the interplay between next-token prediction and the decoding algorithms on the one hand and different end goals on the other. Adopting an asymptotic viewpoint, we find that many of the decoding algorithms explored are not consistent for a vast majority of goals and probability distributions. Further, we find evidence that we might be asking too much of our the decoding algorithms.

Each goal we explore has one class (deterministic vs. stochastic) of decoding algorithms that, except for degenerate cases, are not consistent. What is interesting is that they flip depending on the goal. The N-gram Hamming loss can be thought of as trying to be "correct"; you need to always guess the right answer if possible, hence why randomness hurts. However, when trying to mimic a distribution, this randomness is necessary. A practical implication of our insights is that the user intent should determine the decoding strategy to be used at test time.

We believe there is a lot of interesting future work to be done in this area. One can go deeper into theoretical analysis for these or other loss functions. We particularly found theoretical analysis of the N-gram Hamming loss to be difficult due to the limited methods that can be used to manipulate indicator functions. Future work could also make use of more domain specific assumptions on probability distributions, such as power laws. Other work can include investigating other decoding algorithms, such as Top-K or nucleus sampling, or go even deeper into a specific decoder, such as temperature-scaled random sampling. One can also make the role of stochastic gradient descent more explicit in the training of next-token prediction and investigate if there are any differences that this could cause.

CHAPTER 4

Characterizing the Multiclass Learnability of Forgiving 0-1 Loss Functions

In this chapter we will give a characterization of the learnability of forgiving 0-1 loss functions in the multiclass setting with effectively finite cardinality of the output and label space. To do this, we create a new combinatorial dimension that is based off of the Natarajan Dimension [Natarajan, 1989] and we show that a hypothesis class is learnable in our setting if and only if this Generalized Natarajan Dimension is finite. We also show how this dimension characterizes other known learning settings such as a vast amount of instantiations of learning with set-valued feedback and a modified version of list learning.

4.1 Introduction

Classification is one of the most common tasks in machine learning. Within classification, there is normally a split between binary classification (only two possible outputs) and multiclass classification (more than two possible outputs). The theoretical analysis of these settings shares the same split. In binary classification and multiclass classification, the 0-1 loss has been extensively studied and characterized [Valiant, 1984, Ben-David et al., 1995, Brukhim et al., 2022]. In binary classification, there are 16 possible loss functions ℓ such that $\ell(y, y') \in \{0, 1\}$, but only two where $\ell(y_1, y_2) \neq \ell(y_1, y_2)$ and $\ell(y_2, y_1) \neq \ell(y_2, y_2)$. Thus, a majority of these loss functions are uninteresting and the two that are interesting are the opposites of each other (the 0-1 loss and $1 - \ell_{0-1}$). However, in classification for an output space with cardinality k and a label space with cardinality t, there are $(2^t)^k$ different 0-1 loss functions. Most of these loss functions take the form of more "forgiving" losses (i.e. $\ell(z, y) \in \{0, 1\}$, but there can exist many z_i, y_j such that $\ell(z_i, y_j) = 0$). Even more so, many of these settings can have interesting utility; settings such as paraphrase generation in natural language processing [Zhou and Bhat, 2021], thresholding a metric, ranking with partial feedback [Raman et al., 2023], classifying graphs up to isomorphisms (such as is done for drug discovery [Yang et al., 2024]), and many others can all be seen as allowing for some

tolerance on the output. This chapter attempts to characterize the learnability of a large class of learning problems with these "forgiving" multiclass 0-1 loss functions.

The contributions of this chapter are as follows:

1. We give a new combinatorial dimension based off the Natarajan Dimension [Natarajan, 1989], which we call the Generalized Natarajan Dimension.

2. We show the Generalized Natarajan dimension characterizes learnability of for forgiving 0-1 loss functions in the multiclass setting with minimal extra assumptions on the loss function. We also show that the Generalized Natarajan dimension is incomparable to other known dimensions in the classification literature.

3. We show our characterization implies characterization of various other settings seen in the machine learning literature.

The chapter is organized as follows. Section 4.2 discusses the necessary background information. Section 4.3 details the problem set-up. Section 4.4 shows the results needed to characterize the forgiving 0-1 loss functions along with how the Generalized Natarajan Dimension relates to other dimensions in the literature. Section 4.5 gives concrete applications of our characterization. Finally, Section 4.6 concludes and discusses possible future work.

4.1.1 Related Works

Under the PAC-learning model, binary classification learnability under the 0-1 loss is known to be characterized by the VC-dimension [Vapnik and Chervonenkis, 1974, Shalev-Shwartz and Ben-David, 2014b]. For multiclass classification, there has also been a further split between finite and infinite label cases. For example, it is known that Empirical Risk Minimization (ERM) is a valid learner in the finite label case [Shalev-Shwartz and Ben-David, 2014b], however, Daniely et al. [2015] has shown that not every ERM is a learner in the infinite label case. The Natarajan Dimension has been known to characterize the learnability of the 0-1 loss in the finite label case for many years now [Natarajan, 1989, Ben-David et al., 1995], but only relatively recently has the DS-Dimension been shown to characterize the learnability of the 0-1 loss under the infinite label case [Daniely and Shalev-Shwartz, 2014, Brukhim et al., 2022].

For related work around PAC-learning general loss functions, Ben-David et al. [1995] show the finiteness of the Natarajan dimension is sufficient of learnability of any finite-label multiclass loss function. David et al. [2016] show that sample compression and multiclass learnability are equivalent, but stop short of creating any combinatorial dimension to characterize learnability. Hopkins et al. [2022] show conditions to go from realizable learning

to agnostic learning. However, they require assumptions on loss functions that we do not assume. Recently, Hanneke et al. [2025] were able to upgrade Hopkins et al. [2022] to give an agnostic-to-realizable reduction for all multiclass loss functions with $\{0, 1\}$ output, however, their rates are of the order of $\frac{1}{\epsilon^3}$ in our setting.

Other ways people have generalized PAC-learnability deal with altering the definition. Alon et al. [2022] and Kalavasis et al. [2022] relax PAC-learnability to allow for partial concept classes, which allows the hypotheses to be undefined on parts of the input space. Bressan et al. [2025] alter realizable PAC-learnability to allow for more relaxed errors. We can see our realizable setting is similar to what they study when they have $z = 0$ and $w = \ell$. Using the reduction from Hanneke et al. [2025], one does get a characterization similar to the one arrived at in this chapter. There are two things to note though. One is that their rates have at least an multiplicative factor of $\frac{1}{\epsilon^3}$, which is worse than ours of $\frac{1}{\epsilon^2}$. Second, is they assume there exists a hypothesis that, with probability 1, outputs the correct label. Our realizable case only assumes the infimum of the loss on our hypothesis class is 0. Thus, their assumption is a noticeably stronger assumption than ours. We also show their dimension is incomparable with ours; there exists hypothesis classes where our dimension and theirs can be arbitrarily (and in some cases infinitely) larger than each other. Thus, even if their characterization along with the agnostic-to-realizable reduction did characterize our setting, our dimension is a more clean, precise, and interpretable characterization of the setting. In their paper they state "it would be interesting to find characterizations beyond the J-cube dimensions that allow to fully recover known multiclass sample complexity bounds", the upper bound of which we do recover in our setting.

We also note the similarities of our setting to the setting of list learning. List learning is when the algorithm is allowed to output a list of possible outputs and the loss is given by whether or not the label is present in the learner's outputted list. We can see that, given a multiclass hypothesis h and a loss function ℓ whose output is in $\{0, 1\}$, we can create an equivalent h' and ℓ' where $h'(x) = \{y \mid \ell(h(x), y) = 0\}$ and

$$\ell'(\{y_1, \ldots, y_k\}, y) = \begin{cases} 0 & y \in \{y_1, \ldots, y_k\} \\ 1 & \text{otherwise} \end{cases}.$$

This shows that our setting is equivalent to a modified version of list-learning. List learning was originally studied in Brukhim et al. [2022] and was shown to be characterized by the k-DS dimension when output list size is bounded by k [Charikar and Pabbaraju, 2023]. However, we note the k-DS dimension does not necessarily characterize our setting. In their setting, one is allowed to create a list learner that outputs *any* bounded list and then it is

compared to the best 0-1 learner in the hypothesis class ($\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ but $\mathcal{A}(S) \in \left(2^{\mathcal{Y}}\right)^{\mathcal{X}}$). In our setting, we have to compare to other list learners, not 0-1 learners. We can also only output lists that correspond to sets $\{y' \mid \ell(y, y') = 0\}$. Hanneke et al. [2024] do show that agnostic/realizable list learning and uniform convergence are all equivalent in the setting where the the algorithms output is being compared to list learners whose lists are size k. However, they still allow the algorithm to output any list and our lists are not constrained to output a list of size exactly k. Thus, while there is this connection between the current list-learning literature and our setting, they are distinct settings.

For learning with set-valued labels, Liu and Dietterich [2014] have studied a similar problem where there is a list with a "true" label and other "distractor" labels. They see how well ERM works when trying to find the hypothesis that predicts the true label well. Our setting, where any value in our label set is considered a "true" label, has been characterized in the online setting [Raman et al., 2024b]. The batch setting characterization, however, has remained open as far as we are aware.

4.2 Background and Notation

We will first set up the notion of learnability, which is known as *Probably Approximately Correct* (PAC) learnability [Valiant, 1984].

Definition 4.2.1 (Agnostic PAC-learnability)**.** *Let $\mathcal{X}$ be an input space, $\mathcal{Z}$ be our output space, $\mathcal{Y}$ be our label space, $\mathcal{H} \subset \mathcal{Z}^{\mathcal{X}}$ be a hypothesis class, and $\ell : \mathcal{Z} \times \mathcal{Y}$ be a loss function. Then, $\mathcal{H}$ is agnostic PAC-learnable with respect to ℓ if there exists a learning algorithm $\mathcal{A}$ and a sample function $m : \mathbb{R} \times \mathbb{R} \to \mathbb{N}$ such that $\forall \epsilon, \delta > 0$, for every distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, if we have samples S generated from $\mathcal{D}$, $|S| \geq m(\epsilon, \delta)$, then we have:*

$$\mathbb{P}\left(\mathbb{E}_{\mathcal{D}}\left[\ell(\hat{h}(x), y)\right] \leq \inf_{h \in \mathcal{H}} \mathbb{E}_{\mathcal{D}}\left[\ell(h(x), y)\right] + \epsilon\right) \geq 1 - \delta$$

where $\hat{h} = \mathcal{A}(S) \in \mathcal{H}$.

When $\inf_{h \in \mathcal{H}} \mathbb{E}_{\mathcal{D}}\left[\ell(h(x), y)\right] = 0$, it is called the *realizable* case. We believe this is the best way to generalize the realizable case to our setting as having to compete against a function that has 0 loss is what matters, not competing against a function that predicts the correct label. Were we to do the latter, there can exist cases where $\exists h \in \mathcal{H}$ such that $\mathbb{E}_{\mathcal{D}}\left[\ell(h(x), y)\right] = 0$, but we would not be in the realizable case. For example, let us have $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ and $\ell(\cdot, \cdot) = 0$. Then, for a distribution $\mathcal{D}$ on $\mathcal{X}$ where then our samples are of the form $(x, c(x))$ where $c \in \mathcal{Y}^{\mathcal{X}} \setminus \mathcal{H}$ we can see this would be in the realizable case for our

setting, but not for the stricter version.

One of the most important loss functions is the 0-1 loss function [Wang et al., 2022] and it is a common place to start when studying a new setting (or its equivalent in the setting) [Shalev-Shwartz and Ben-David, 2014b, Brukhim et al., 2022, Charikar and Pabbaraju, 2023, Guermeur, 2007]:

$$\ell_{0-1}(y, y') = \begin{cases} 0 & y = y' \\ 1 & \text{otherwise} \end{cases}$$

For multiclass classification with finite labels using the 0-1 loss, it is known that this setting is characterized by the Natarajan dimension [Natarajan, 1989, Shalev-Shwartz and Ben-David, 2014b]:

Definition 4.2.2 (Natarajan Dimension)**.** *We say that a hypothesis class $\mathcal{H}$ Natarajan shatters a set $S = \{s_1, ..., s_n\}$ if $\exists h_1, h_2 \in \mathcal{H}$ such that*

1. *$\forall s_i \in S$, $h_1(s_i) \neq h_2(s_i)$*
2. *$\forall S' \subseteq S$ $\exists h \in \mathcal{H}$ where $\forall s \in S'$ $h(s) = h_1(s)$ and for all $s \in S \setminus S'$ $h(s) = h_2(s)$*

The Natarajan dimension of a hypothesis class $\mathcal{H}$, denoted $Ndim(\mathcal{H})$, is the cardinality of the largest set that the hypothesis class Natarajan shatters.

One nice property of the 0-1 loss function is the *identity of indescernibles*.

Definition 4.2.3 (Identity of Indiscernibles)**.** *A loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ has the identity of indiscernibles property if*

$$\forall y_1, y_2 \in \mathcal{Y}, \quad \ell(y_1, y_2) = 0 \iff y_1 = y_2$$

This property is seen in many well-used loss functions such as the 0-1 loss and the squared loss. Hopkins et al. [2022] show that this property can be used to create an agnostic learner from a realizable learner and has been used to characterize learnability problems [Raman et al., 2024a,b]. In fact, Hopkins et al. [2022] prove this result by relating the loss between $h(x)$ and $h'(x)$ to the classification error $\mathbb{P}(h(x) \neq h'(x))$. Thus, just like we saw in the discussion of the definition the realizable setting, this notion of equality of labels and loss values has been extremely interlinked when the loss function satisfies the identity of indiscernibles.

In this work we will explicitly **not** assume the identity of indiscernibles or any generalization of it to different output and label spaces. In fact, our assumptions assume hardly any structure at all, allowing our results to generalize to many settings.

We shall now introduce some notation. Given a loss function $\ell : \mathcal{Z} \times \mathcal{Y} \to \{0, 1\}$, let $C := C(\ell) = \{(z, y) \mid \ell(z, y) = 0\}$. We will refer to this the equality set. We also define $\sigma(z) := \sigma_\ell(z) = \{y \mid \ell(z, y) = 0\}$. Thus, $\sigma(z)$ is all the labels y where (z, y) achieves 0 loss. We similarly define $\tau(y) := \tau_\ell(y) = \{z \mid \ell(z, y) = 0\}$. We will oftentimes omit the ℓ subscript from σ, τ and ℓ parameter for C, but there is always an implicit loss function these are referring to. Further, notice that from any one of C, ℓ, σ, or τ, we can reconstruct the others.

Note how σ, τ create equivalence relations on $\mathcal{Z}$ and $\mathcal{Y}$ by $z_1 \sim_\sigma z_2 \iff \sigma(z_1) = \sigma(z_2)$ and similar for $\mathcal{Y}$ using τ. Let $\sigma(\mathcal{Z})$ and $\tau(\mathcal{Y})$ be these equivalence classes. We note that $|\sigma(\mathcal{Z})| < \infty \implies |\tau(\mathcal{Y})| < 2^{|\sigma(\mathcal{Z})|}$ and $|\tau(\mathcal{Y})| < \infty \implies |\sigma(\mathcal{Z})| < 2^{|\tau(\mathcal{Y})|}$.

4.3 Setup

Let us have an output space $\mathcal{Z}$ and label space $\mathcal{Y}$. We analyze loss functions ℓ that satisfy the following constraints:

- $\ell : \mathcal{Z} \times \mathcal{Y} \to \{0, 1\}$.
- $|\sigma(\mathcal{Z})| < \infty$ and $\forall z \in \mathcal{Z}$, $|\sigma(z)| < \infty$
- $\forall z_1, z_2 \in \mathcal{Z}$, $\sigma(z_1) \not\subset \sigma(z_2)$. We note $\subset$ means *strict* subset; we do allow equalities.

Hopkins et al. [2022] have showed that one can have a realizable learner that is not agnosticly learnable in multiclass settings that do not have the identity of indiscernibles. The example they show relies on the loss function taking 3 values, 0, 1, and c. Thus, assumption #1 allows our loss functions can only take values of 0 or 1, and our results show that this is sufficient to have both agnostic and realizable learnability.

Assumption #2 is needed to allow our generalization of the Natarajan dimension to work. The Natarajan dimension characterizes multiclass learnability for finite output spaces. We can generalize this to infinite output and label spaces, so long as σ partitions our output set into a finite amount of partitions and each output has 0 loss on a finite amount of labels. Since our loss can only distinguish over equivalence classes, this can be thought of as bounding the "effective" output and label space sizes. We will refer to this as *effectively finite*.

Assumption #3 is here as the problem does not make intuitive sense without it. The goal of our learning problem is to minimize the loss. Since we have full access to the loss values, if the outputs that z_1 takes no loss on is strictly contained in the outputs z_2 takes no loss on (i.e. $\sigma(z_1) \subset \sigma(z_2)$), why would one ever want to output z_1? It would always be

objectively better to output z_2. Thus, without loss of generality, we assume this does not happen, otherwise we can just replace all instances of z_1 with z_2 in our hypothesis class. In our characterization, this assumption is used in the proof that the finiteness of our dimension is necessary for learning. In Appendix C.4.1 we show a counterexample to our main theorem of characterization if this assumption is broken as well.

4.3.1 Example: Multilabel Ranking

A ranking learning problem is one where a learner is required to output a permutation of $[k]$ that indicates a ranking of k elements. An example could be outputting what a learner believes to be the ranking of a person's favorite movies. However, one might only care about whether or not the learner got the person's top 10 movies correct. Thus, while there are a total of K outputs, the quality of the output need not depend on all K values, but only on the top $p \leq K$. For more information on multilabel ranking, we refer the readers to Raman et al. [2023].

For this example, we will assume the output and label space are the same. Given an input space $\mathcal{X}$, output/label space $\mathcal{Y}$ (permutations of $[K]$), and a hypothesis class $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$, we can get the following losses:

$$\ell_{0-1}^{sum@p}(y, y') = \begin{cases} 0 & \forall i \in [p] \quad y_i = y'_i \\ 1 & \text{otherwise} \end{cases}$$

and

$$\ell_{0-1}^{prec@p}(y, y') = \begin{cases} 0 & \{y_1, \ldots, y_p\} = \{y'_1, \ldots, y'_p\} \\ 1 & \text{otherwise} \end{cases}.$$

The $\ell_{0-1}^{sum@p}$ loss cares about if the ranking is in the correct exact order for the top p values. Meanwhile, the $\ell_{0-1}^{prec@p}$ loss only cares about if the values in the top are correct and order does not matter. These losses then create equivalence classes over our output/label space by whether the top p are in correct order for $\ell_{0-1}^{sum@p}$ or whether the top p are the correct set for $\ell_{0-1}^{prec@p}$. Further, no one ranking output dominates another output since they only get 0 loss on the values in their equivalence class. Thus, we can see this setting will match the assumptions of our learning problem.

In Section 4.5.1.2 we show a characterization of this setting along with the characterization of a more general version of the above that allows for the labels to be relevance scores of labels and not themselves a ranking. This in turn, by the results of Raman et al. [2023], show that our characterization applies to many well-used multilabel ranking setups.

4.4 The Generalized Natarajan Dimension

With spaces $\mathcal{Z}$, $\mathcal{Y}$ and loss function ℓ as defined above, the goal of this chapter is to characterize PAC-learnability of a hypothesis class.

First, notice if $z \sim_\sigma z'$ and $y \sim_\tau y'$, then

$$\ell(z, y) = \ell(z', y) = \ell(z', y') = \ell(z, y')$$

This shows that our loss function respects our equivalence relation, and thus $\exists \ell^{\sigma,\tau} : \sigma(\mathcal{Z}) \times \tau(\mathcal{Y}) \to \{0, 1\}$ that agrees with ℓ. Thus:

Corollary 4.4.0.1. *Let $\mathcal{D}$ be a distribution on $\mathcal{X} \times \mathcal{Y}$, and $\mathcal{D}^{\sigma,\tau}$ be its quotient onto $\mathcal{X} \times \tau(\mathcal{Y})$. Further, let $h \in \mathcal{Z}^{\mathcal{X}}$, and h^σ be its quotient onto $(\sigma(\mathcal{Z}))^{\mathcal{X}}$. Then*

$$\mathbb{E}_{\mathcal{D}}[\ell(h(x), y)] = \mathbb{E}_{\mathcal{D}^{\sigma,\tau}}[\ell^{\sigma,\tau}(h^\sigma(x), y)]$$

One can also very easily convert a $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem into a $(\mathcal{X}, \sigma(\mathcal{Z}), \tau(\mathcal{Y}), \sigma \circ \mathcal{H}, \ell^{\sigma,\tau})$ learning problem by creating the projection map $p : \mathcal{Z} \to \sigma(\mathcal{Z})$, and then using $p \circ h$ for predictions. This way it is also very easy to recover the original $h \in \mathcal{H}$. This can also be easily implemented in code by replacing the quotient spaces $\sigma(\mathcal{Z}), \tau(\mathcal{Y})$ with the set of representatives for the equivalence classes and then mapping each element in $\mathcal{Z}, \mathcal{Y}$ to its representative. Thus, we can see that the learnability of $(\mathcal{X}, \sigma(\mathcal{Z}), \tau(\mathcal{Y}), \sigma \circ \mathcal{H}, \ell^{\sigma,\tau})$ implies the learnability of $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ while only using assumptions #1 and #2.

We also note that we use $(\mathcal{X}, \mathcal{Z}^C, \mathcal{Y}^C, \mathcal{H}^C, \ell^C)$ as alternative notation to $(\mathcal{X}, \sigma(\mathcal{Z}), \tau(\mathcal{Y}), \sigma \circ \mathcal{H}, \ell^{\sigma,\tau})$. When the context is clear, we will drop superscripts and abuse notation between the equivalence classes versions of $\mathcal{Z}$, $\mathcal{Y}$, $\mathcal{H}$, ℓ and themselves.

Now, we introduce our new dimension, the Generalized Natarajan dimension, based off Natarajan [1989].

Definition 4.4.1 (Generalized Natarajan Dimension)**.** *We say that a hypothesis class $\mathcal{H}$ and loss function ℓ Generalized Natarajan shatters a set $S = \{s_1, ..., s_n\}$ if $\exists h_1, h_2 \in \mathcal{H}$ such that*

1. *$\forall s_i \in S$, $\sigma(h_1(s_i)) \neq \sigma(h_2(s_i))$*
2. *$\forall S' \subseteq S$ $\exists h \in \mathcal{H}$ where $\forall s \in S'$ $\sigma(h(s)) = \sigma(h_1(s))$ and for all $s \in S \setminus S'$ $\sigma(h(s)) = \sigma(h_2(s))$*

The Generalized Natarajan dimension of a hypothesis class $\mathcal{H}$, denoted $GNdim(\mathcal{H}, \ell)$, is the cardinality of the largest set that the hypothesis class generalized Natarajan shatters.

From this we have the immediate corollary:

Corollary 4.4.0.2. *Given a hypothesis class $\mathcal{H}$ and loss function ℓ as studied in this chapter, we have*

$$GNdim(\mathcal{H}, \ell) = Ndim(\sigma \circ \mathcal{H}) = GNdim(\sigma \circ \mathcal{H}, \ell^{\sigma,\tau})$$

The Generalized Natarajan Dimension effectively changes what it means to be equal. One implicit property relied on when a loss function has the identity of indiscernibles is the relationship between equality and loss value. Given $\mathcal{Z} = \mathcal{Y}$, we have that $y_1 = y_2$ if and only if $\ell(y_1, y_2) = 0$ and $y_1 \neq y_2$ if and only if $\ell(y_1, y_2) > 0$. This allows us to use equality of our labels to infer how the loss will act. This is now no longer the case; we can have $y_1 \neq y_2$ but still have $\ell(y_1, y_2) = 0$. Because of this, our dimension needs to explicitly use the loss values instead of being able to use the labels by proxy.

In Theorem 4.4.1 we show the Generalized Natarajan dimension characterizes our setting. Thus, the proxy for equality needed is for the equivalency of the set of labels where z_1 and z_2 achieve 0 loss for (i.e. $\sigma(z_1) = \sigma(z_2)$). This is a very strict type of equality. One can think of a setting where $|\mathcal{Z}|$ is very large (but finite) with $\sigma(z_1)$ and $\sigma(z_2)$ being almost equal. However, since they are not exactly equal, this dimension does not get any smaller than the Natarajan dimension. This is quite counter-intuitive as one would expect learnability to be much easier in this setting due to the vast amount of values that will output 0 loss, but it is not as you can always make a distribution that puts a lot of mass where $\sigma(z_1)$ and $\sigma(z_2)$ differ.

4.4.1 Characterizing Learnability of Forgiving 0-1 Loss Functions

With this setup, we state our main theorem:

Theorem 4.4.1. *A $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem is PAC-learnable if and only if $GNdim(\mathcal{H}, \ell) < \infty$.*

We prove this through showing the necessity of $GNdim(\mathcal{H}, \ell) < \infty$ in Lemma 4.4.2 and then showing sufficiency in Lemma 4.4.3.

The only if direction is proven through an alteration of the No-Free-Lunch Theorem [Shalev-Shwartz and Ben-David, 2014b]. The if direction is done through a reduction into the $(\mathcal{X}, \sigma(\mathcal{Z}), \tau(\mathcal{Y}), \mathcal{H}, \ell^{\sigma,\tau})$ learning problem. Once this reduction is done, we can bound the the VC-dimension of the loss class by the Generalized Natarajan dimension. This then shows that ERM is a learner and gives an upper bound on the sample complexity.

Lemma 4.4.2. *Let us have a learning problem $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ as has been studied in this chapter. Then, $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ is PAC-learnable $\implies$ the Generalized Natarajan Dimension of $\mathcal{H}$ is finite.*

Proof Sketch. We show that $GNdim(\mathcal{H}, \ell) = \infty \implies (\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ is not learnable. To do this, we modify the No Free Lunch Theorem to change the realizable distributions we are looking over. In the No Free Lunch Theorem for the 0-1 loss, we have a shattered set $\mathcal{X}$ and create distributions over $\mathcal{X}$ where the labels are labeled by the $h_i \in \mathcal{H}$ that witness the shattering of $\mathcal{X}$. This does not work in our setting for two reasons: one is that our output space and label space need not be the same and the second is, even if they were, we can have $h_i(x) \neq f(x)$ but still have $\ell(f(x), h_i(x)) = 0$. To fix this, we change our realizable distribution to be uniform over the outputs in $\sigma(h_i(x))$ to make sure if $h, h' \in \mathcal{H}$, $\sigma(h(x)) \neq \sigma(h'(x))$, then $\exists y$ such that $\ell(h(x), y) \neq \ell(h'_c(x), y)$. The rest of the proof is then modified to work with these changes. □

We give the full proof of the above in Appendix C.2.

Since we have shown necessity the of $GNdim(\mathcal{H}, \ell) < \infty$, we now show sufficiency.

Lemma 4.4.3. *Let us have a learning problem $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ as has been studied in this chapter. Then, $GNdim(\mathcal{H}, \ell)$ is finite $\implies$ Learnability of $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$.*

We give two proofs of this lemma in Appendix C.3. Appendix C.3.1 uses the VC-dimension of the loss class and Appendix C.3.2 shows a different proof of a special case of when $\mathcal{Z} = \mathcal{Y}$. While this specialized proof also gives a worse sample complexity bound than the VC-dimension proof, it is included as we believe a direct proof through uniform convergence adds intuition.

Given lemmas 4.4.2 and 4.4.3, we have now proved Theorem 4.4.1. We note the sufficiency proof does not rely assumptions #3. Therefore, for any effectively finite learning problem with a loss function whose output is in $\{0, 1\}$, we have that finite $GNdim(\mathcal{H}, \ell)$ is sufficient for learnability.

To find sample complexity, we use the VC-dimension of the loss class. Since we have now shown that $GNdim(\mathcal{H}, \ell)$ characterizes our setting, Lemma C.1.1 shows that the VC-dimension of the loss class does as well. From this, we use the well-known sample complexity bounds of binary classification get the following result:

Corollary 4.4.3.1. *Given an $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem studied in this chapter, we have the following bounds on the agnostic PAC-learning sample complexity:*

$$m(\epsilon, \delta) \leq O\left(\frac{GNdim(\mathcal{H}, \ell) \log\left(|\sigma(\mathcal{Z})|\right) + \log(1/\delta)}{\epsilon^2}\right).$$

Bressan et al. [2025] state that a class is not learnable when there is at least a pair of outputs that are difficult to distinguish. Our results show that, while true, this does not tell the full picture. Two outputs being indistinguishable is also the only way we can get a smaller upper bound for the sample complexity as well. In fact, over all learning settings that satisfy our assumptions, the only way we can have our hypothesis class go from unlearnable to learnable is through changing the loss function to have more equivalent outputs.

However, this can be a blessing or a curse. If a loss function ℓ and hypothesis class $\mathcal{H}$ pair has Generalized Natarajan dimension d, then creating an ℓ' such that $C(\ell) \subset C(\ell')$ can lead to a larger Generalized Natarajan dimension, and thus possibly a larger sample complexity required to learn. We show an example of this, albeit contrived, in the proof of Proposition 10.

We can see from above then how the "forgiveness" of a loss function is hypothesis class-dependent. We also see that, for a given hypothesis class, all loss functions that induce the same equivalence classes on $\mathcal{Z}$ will have the same Generalized Natarajan dimension. For example, if a loss does not induce any equivalent outputs, then, by Corollary 4.4.0.1, we have that our Generalized Natarajan dimension of this hypothesis class/loss pair is equal to the Natarajan dimension of the hypothesis class. Thus, even if a loss function looks to be more forgiving, in terms of learnability it might not be. Therefore, the colloquial meaning of a "forgiving" loss function and what loss functions are truly more forgiving are at odds with each other.

4.4.2 Relation to Other Dimensions

In Corollary C.1.1.1 of the appendix we show that the VC-dimension of the loss class also characterizes our setting. The fact that the Generalized Natarajan dimension and the VC-dimension of the loss class both rely on both the hypothesis class and the loss function is crucial to them being able to characterize the setting. Below, we show that any dimension that relies solely on comparing hypotheses through the normal notions of "shattering" will not characterize our setting.

Here we give the incomparability results of the Natarajan Dimension and the d_J dimension [Bressan et al., 2025].

Proposition 9. *Let $\mathcal{X}$ be our input space and $\mathcal{Y}$, $|\mathcal{Y}| < \infty$ be our label space. There exists a hypothesis class $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ such that $\forall J \subset \mathcal{Y}$, $Ndim(\mathcal{H}) = d_J(\mathcal{H}) = \infty$, but there exists a loss function ℓ such that $GNdim(\mathcal{H}, \ell) = 0$.*

Proof. Let $|\mathcal{X}| = \infty$, $\mathcal{H} = \mathcal{Y}^{\mathcal{X}}$ and let $\ell(y, y') = 0$ for all $y, y' \in \mathcal{Y}$. Note that, since $\mathcal{H}$ is all functions from inputs to outputs, then, $Ndim(\mathcal{H}) = \infty$, and $\forall J \subset \mathcal{Y}$, $d_J(\mathcal{H}) = \infty$. However,

since our loss partitions $\mathcal{Y}$ to all the same equivalence class, $\sigma \circ \mathcal{H}$ contains only 1 element, thus $GNdim(\mathcal{H}, \ell) = 0$. □

Proposition 10. *For all $q \in \mathbb{N}$, there exists a finite input space $\mathcal{X}$, finite output space $\mathcal{Y}$, finite hypothesis class $\mathcal{H}$, and loss function ℓ such that $Ndim(\mathcal{H}) = 1$, and $\forall J \subset \mathcal{Y}$, $d_J(\mathcal{H}) \leq 1$, but $GNdim(\mathcal{H}, \ell) = q$.*

Proof. Let $q \in \mathbb{N}$, $\mathcal{X} = \{x_1, \ldots, x_q\}$, $\mathcal{Y} = \{y_1, \ldots, y_{q^{q+1}}\}$, let

$$h_i(x_j) = y_{q(i-1)+j},$$

and let $\mathcal{H} = \{h_i \mid i \in [q^q]\}$. Notice how $Ndim(\mathcal{H}) = 1$ as no hypotheses have any overlapping outputs. In Bressan et al. [2025], they state that $Ndim(\mathcal{H}) \geq \max_J d_J(\mathcal{H})$, thus $\forall J \subset \mathcal{Y}$, $d_J(\mathcal{H}) \leq 1$. Now, let $\mathcal{H}' = [q]^{\mathcal{X}}$. Note there are $|\mathcal{H}'| = q^q$, thus create an arbitrary bijection $f : \mathcal{H}' \to \mathcal{H}$. Partition $\mathcal{Y}$ on the following:

$$y \sim y' \iff \exists h', h^\dagger \in \mathcal{H}', \exists x_i \in \mathcal{X} \text{ where } h'(x_i) = h^\dagger(x_i) \text{ and } f(h')(x_i) = y, f(h^\dagger)(x_i) = y'$$

Notice this partitions $\mathcal{Y}$ such that each partition is equivalent to a specific $c \in [q]$ for the outputs of $\mathcal{H}'$. Thus, let

$$\ell(y, y') = \begin{cases} 0 & y \sim y' \\ 1 & \text{otherwise} \end{cases}.$$

From the definition of the equivalence class, we can see that $\sigma \circ \mathcal{H}$ is then equivalent to all functions from $\{x_1, \ldots, x_q\}$ to $[q]$, which has a Natarajan dimension of q. □

We do note that, due to our characterization results along with results in Ben-David et al. [1995] and Bressan et al. [2025], we have that $GNdim(\mathcal{H}, \ell) = \infty \implies Ndim(\mathcal{H}) = \infty \implies \max_J d_J(\mathcal{H}) = \infty$. Thus, while the Generalized Natarajan dimension can be be infinitely smaller and arbitrarly larger than these dimensions, it can not be infintely larger.

Since the Natarajan dimension characterizes the 0-1 loss in our setting, is is not surprising to see cases where $GNdim(\mathcal{H}, \ell) < Ndim(\mathcal{H})$, but it might be surprising to see $GNdim(\mathcal{H}, \ell) > Ndim(\mathcal{H})$. However, we believe that looking through the angle of *discernment* gives the intuition for why this can happen. The job of the algorithm is to, through the samples given, discern which hypothesis are useful and which are not. Forgiving losses, while bringing the actual loss values of all hypotheses down, does not need to add information on which hypothesis might be better than others. For example, it might help some hypotheses more than others, but the only way to tell is to find points on which these hypotheses differ. Now though, this job is harder since hypotheses that did disagree before now agree on more

places. Since the algorithm needs to find a hypothesis that is close to the infimum, even though all loss values are lowered, due to this difficulty in discernment it can be harder to find such a hypothesis.

We also note that the same proof examples and ideas as above can be used to show the incomparability of the Generalized Natarajan dimension to the DS-dimension [Daniely and Shalev-Shwartz, 2014, Brukhim et al., 2022] and the k-DS-dimension [Charikar and Pabbaraju, 2023].

Corollary 4.4.3.2. *There exist input spaces $\mathcal{X}, \mathcal{X}'$, output spaces $\mathcal{Y}, \mathcal{Y}'$, hypothesis classes $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}, \mathcal{H}' \subset \mathcal{Y}'^{\mathcal{X}'}$, and loss functions ℓ, ℓ' such that $GNdim(\mathcal{H}, \ell) = 0$ and $DS(\mathcal{H}) =k$-$DS(\mathcal{H}) = \infty$ and $GNdim(\mathcal{H}', \ell') = q \in \mathbb{N}$ and $DS(\mathcal{H}') = k$-$DS(\mathcal{H}') = 0$ for any $k \in \mathbb{N}$, $k \geq 1$.*

One can see that $Ndim(\mathcal{H}) = \infty \implies DS(\mathcal{H}) = \infty$ since all distributions over $\mathcal{X}$ and a finite subset of $\mathcal{Y}$ is included is a subset of distributions over all $\mathcal{X} \times \mathcal{Y}$. Thus, we have as well $GNdim(\mathcal{H}, \ell) = \infty \implies DS(\mathcal{H}) = \infty$. It is known that for any $k < k'$, k-$DS(\mathcal{H}) >k'$-$DS(\mathcal{H})$ [Charikar and Pabbaraju, 2023], thus we posit a conjecture that $GNdim(\mathcal{H}, \ell) = \infty$ does **not** imply k-$DS(\mathcal{H}) = \infty$ for $k > 1$. This conjecture comes from the ideas stated in Section 4.1.1 where the modified list-learning setting can be both easier and harder than the normal list-learning setting.

4.5 Applications of Characterization

Given the minimal assumptions made on our learning setting, the Generalized Natarajan dimension is able to characterize many machine learning scenarios. In this section we detail some of these settings.

4.5.1 Characterizing Learnability of Set Learning

We define set learning as receiving set-valued feedback and checking whether or not the output is in the set. More precisely, let $\mathcal{X}$ be our input space, let $\mathcal{Z}$, $2 < |\mathcal{Z}| < \infty$, be our output space, let our label space be $\mathcal{Y} \subset 2^{\mathcal{Z}}$, such that no $y, y' \in \mathcal{Y}$ have $y \subseteq y'$ or $y' \subseteq y$, and let us have a hypothesis class $\mathcal{H} \subset \mathcal{Z}^{\mathcal{X}}$. The samples we receive are of the form $(x, v) \in \mathcal{X} \times \mathcal{Y}$. Finally, our loss function becomes for any $z \in \mathcal{Y}$ and for any $y \in \mathcal{Y}$:

$$\ell(z, y) = \begin{cases} 0 & z \in y \\ 1 & \text{otherwise} \end{cases}$$

We give an example of what this loss function looks like for $\mathcal{Z} = \{1, 2, 3\}$ and $\mathcal{Y} = 2^{\mathcal{Z}} \setminus \{\emptyset\}$ in Figure 4.1.

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccc} \{1\} & \{2\} & \{3\} & \{1,2\} & \{1,3\} & \{2,3\} & \{1,2,3\} \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Figure 4.1: An example loss matrix for the set learning setup where $\mathcal{Z} = \{1, 2, 3\}$ $\mathcal{Y} = 2^{\{1,2,3\}} \setminus \{\emptyset\}$. Since each $z \in \mathcal{Z}$ is only equivalent to themselves, we see this is characterized by the Natarajan dimension. Note this would still be true if $\mathcal{Y} = 2^{\{1,2,3\}} \setminus \{\emptyset, \{1\}, \{2\}, \{3\}\}$ as well.

label space $\mathcal{Y} = \{0, 1, \ldots, B\}^K$ (denotes how relevant a label is where higher is more relevant), and a hypothesis class $\mathcal{H} \subset \mathcal{Z}^{\mathcal{X}}$, we can create the following loss functions. For a given K and p, let

Note how these problems fit the assumptions in this chapter. Thus, we have the following corollary:

Corollary 4.5.0.1. *For all instantiations of set learning that meet our assumptions, the learnability of the learning problem is characterized by the Natarajan dimension.*

This problem has been characterized in the online setting [Raman et al., 2024b], but for a majority of these instantiations the batch setting has remained open as far as we are aware until now.

This is a very useful result as many problems can be formulated as a set-learning problem. For example, any setting that is classifying up-to equivalence classes is included under this setting. This is because, for any $y, y' \in \mathcal{Y}$, $\ell(y, y') = \ell([y], [y'])$ where $[y], [y']$ are the equivalence classes y, y' are in, respectively. In the following subsections we give two concrete examples.

4.5.1.1 Classifying Graphs up to Isomorphism

Classifying graphs is well-used in fields such as drug discovery [Yang et al., 2024]. In cases such as molecules, it does not matter which graph is outputted, so long as it is isomorphic to the correct label graph. It is known that graph isomorphisms form a partition over the graph space, thus we have the following:

Corollary 4.5.0.2. *Given an input space $\mathcal{X}$ and output/label space of graphs of with finite amount of edges/nodes, the Generalized Natarajan dimension characterizes learnability over any 0-1 loss function that respects graph isomorphisms (i.e. $\forall y, y', y^\dagger \in \mathcal{Y}$, y isomorphic to y', $\ell(y, y^\dagger) = \ell(y', y^\dagger)$).*

4.5.1.2 Multilabel Ranking

Since the example in Section 4.3.1 meets the assumptions of our learning problem, we can see our characterization applies there.

To generalize this example to more real-life scenarios, we note that, as discussed in Raman et al. [2023], labels are normally given by a relevance score, not an exact ranking. A relevance score is an ordered set $y \in \{0, 1, \ldots, B\}^K$ where the i^{th} value denotes how relevant the i^{th} label is (a higher value means more relevant). They create two losses:

$$\ell^{sum@p}(z, y) = \sum_{i=1}^{K} \min\left(z_i, p+1\right) y^i - C_y^p$$

and

$$\ell^{prec@p}(z, y) = \sum_{i=1}^{K} 1_{\{z_i \leq p\}} y^i - C_y^p,$$

where y^i is the relevance score of the label ranked at z_i and the C_y^p values are a normalization constant that depend only on p and y that ensure

$$\min_{z \in \mathcal{Z}} \ell^{prec@p}(z, y) = \min_{z \in \mathcal{Z}} \ell^{sum@p}(z, y) = 0.$$

The loss $\ell^{sum@p}(z, y)$ essentially checks if the values and the order are correct for the top p values and $\ell^{sum@p}(z, y)$ checks whether or not the set of the top p values are correct. From these, we can see there is no z, z' such that $\sigma(z) \subset \sigma(z')$ for either loss.

Raman et al. [2023] also show that, for either of the following sets, that if one loss is learnable, then any loss in that set is learnable:

$$\begin{aligned}
&\mathcal{L}(\ell^{sum@p}) = \\
&\{\ell \in \mathbb{R}^{\mathcal{Z} \times \mathcal{Y}} \mid \ell = 0 \iff \ell^{sum@p} = 0\} \cap \{\ell \in \mathbb{R}^{\mathcal{Z} \times \mathcal{Y}} \mid z \overset{[p]}{=} z' \implies \ell(z, y) = \ell(z', y)\} \\
&\mathcal{L}(\ell^{prec@p}) = \\
&\{\ell \in \mathbb{R}^{\mathcal{Z} \times \mathcal{Y}} \mid \ell = 0 \iff \ell^{prec@p} = 0\} \cap \{\ell \in \mathbb{R}^{\mathcal{Z} \times \mathcal{Y}} \mid z \overset{p}{=} z' \implies \ell(z, y) = \ell(z', y)\}.
\end{aligned}$$

where $z \overset{[p]}{=} z'$ denotes the order of the top p are the same and $z \overset{p}{=} z'$ denotes the set of the top p are the same. Raman et al. [2023] show that many well-used ranking loss functions

are captured in these sets of losses.

Now, let us create the following loss functions:

$$\ell_{0-1}^{sum@p}(z,y) = 1_{\{\ell^{sum@p}(z,y)>0\}}$$
$$\ell_{0-1}^{prec@p}(z,y) = 1_{\{\ell^{prec@p}(z,y)>0\}}.$$

Notice how these are forgiving $0-1$ loss functions that satisfy the assumptions for our learning problem. Thus, the Generalized Natarajan Dimension characterizes these functions. Further, notice $\ell_{0-1}^{sum@p}(z,y) \in \mathcal{L}(\ell^{sum@p})$ and $\ell_{0-1}^{prec@p}(z,y) \in \mathcal{L}(\ell^{prec@p})$. Therefore, we get the following corollary:

Corollary 4.5.0.3. *$GNdim(\mathcal{H}, \ell_{0-1}^{prec@p})$ characterizes all loss functions in $\mathcal{L}(\ell^{prec@p})$ and $GNdim(\mathcal{H}, \ell_{0-1}^{sum@p})$ characterizes all loss functions in $\mathcal{L}(\ell^{sum@p})$.*

This is the first characterization that we are aware of that gives a dimension on the entire hypothesis class and not on individual indices.

4.5.2 Characterizing Learnability of Modified List Learning

As noted in Section 4.1.1, we characterize a modified version of list learning for many possible set-ups. This version is the "flip" of set learning where now our hypotheses output lists and we want to minimize $\mathbb{P}(y \in h(x))$. Thus, given possible labels $\mathcal{Y}$, $|\mathcal{Y}| < \infty$, a set of lists $\mathcal{Z} \subset 2^{\mathcal{Y}}$ where $\forall z, z' \in \mathcal{Z}$, $z \not\subset z'$, hypothesis class $\mathcal{H} \subset \mathcal{Z}^{\mathcal{X}}$ and loss:

$$\ell(\{y_{i_1}, \ldots, y_{i_n}\}, y) = \begin{cases} 0 & y \in \{y_{i_1}, \ldots, y_{i_n}\} \\ 1 & \text{otherwise} \end{cases}$$

for $|i| = n < |\mathcal{Y}|$, this gives us a list learning problem that satisfies our constraints. Thus, we have the following corollary:

Corollary 4.5.0.4. *The version of list learning described above is characterized by the Generalized Natarajan dimension.*

4.6 Conclusion and Future Work

In this chapter we have characterized a large set of more forgiving 0-1 loss functions in the effectively finite label multiclass setting.

Through the properties of our Generalized Natarajan dimension, we can see that the Natarajan dimension will still characterize many of these loss functions as well. Many of

these loss functions could be mostly zeros, but if the quotient space of the outputs is the same as the output space, the learnability is the same as in the 0-1 loss. While this is unintuitive, we believe this is due to how stringent PAC-learnability is. Since PAC-learnability requires learnability over all distributions, one can see how, given a hypothesis class and a loss function that does not reduce the output space, one can adversarially create a distribution that puts a lot of mass on the areas where the sets $\sigma(h(x))$ disagree. This will then nullify the "forgiving" portion of the loss function. Overall, thinking through discernment shows that forgiving losses might or might not actually live up to their name.

Further work can be done by finding a way to remove the assumption on the loss function to allow for outputs where one output is "dominated" by the other. Also, seeing whether or not extensions to effectively infinite output/label spaces would mirror the Natarajan/DS dimension split as in the 0-1 loss case would be an interesting result. Finally, further work of tightening the sample complexity bounds can be useful to give quantitative results on forgiveness. For example, deriving a lower bound might show that forgiveness can help a lot on specific distributions. Also, since our analysis of the upper bound of the sample complexity goes through the VC-dimension of the loss class, there is potential work to show the tightness of this bound.

CHAPTER 5

Conclusion and Discussion

In this thesis we have looked at three settings in learning theory and have proven interesting results in each. As the machine learning field continues to grow, it is important to make sure the theory grows along with it. As this thesis shows, many theoretical questions can have results that have practical implications. By blending practice and theory, we make sure our practical results are not just noise, but legitimate progress.

While each chapter discusses future work in their respective settings, there is much future work to be done in this field as well. There are many interesting questions that arise from current machine learning practices that can be grounded in learning theory. As the work from myself and others in my field have shown, theory can be applied to every aspect of machine learning. I encourage any reader who is interested in applied machine learning research to think about theoretical results in tandem with practical changes. This will make sure the field stays grounded and I believe the insights from one can help progress the other.

Thank you for reading.

APPENDIX A

Appendix For Chapter 2

A.1 Conversion from Scalar Valued Linear Covering Number Bound to Linear Mapping Covering Number Bound

Suppose we have sets $\mathcal{X}, \mathcal{M} \subset \mathbb{R}^d$, where $\forall w \in \mathcal{M}$, $\|w\|_r \leq B_w$ and $\forall x \in \mathcal{X}$, $\|x\|_s \leq B_x$ for some positive values r and s. Suppose we also have a function class $\mathcal{F} = \{x \to w^\top x \mid w \in \mathcal{W}\}$ and a log covering number C for this function class on inputs $\{x_i\}_{i=1}^n \subset \mathcal{X}$. Let $\mathcal{W} \subset \mathbb{R}^{k \times d}$, where $\forall W \in \mathcal{W}$, $\forall i \in [k]$, $\|W_i\|_r \leq B_w$.

Now, given a $W \in \mathcal{W}$, let us choose

$$\hat{W} = \begin{bmatrix} \hat{W}_1 \\ \hat{W}_2 \\ \vdots \\ \hat{W}_k \end{bmatrix} \in \mathcal{W}$$

where $\hat{W}_j^\top$ is the column vector that would be chosen to cover $W_j^\top$ in the scalar case. Then notice for a positive value q and for any $t \in [n]$:

$$\left\|(W - \hat{W})x_t\right\|_q^q = \sum_{j=1}^{k}((W_j - \hat{W}_j)x_t)^q \leq k\epsilon^q$$

Thus we can see

$$\log N_\infty \left(\mathcal{F}, k^{1/q}\epsilon, N, \|\cdot\|_q\right) \leq kC$$

Therefore, if C does not rely on n, neither does this bound.

A.2 Proof of Lemma 2.3.2

Notice how the right hand side is a lower bound for the left hand side. Thus, we need to show $\mathcal{N}_\infty\left(\mathcal{F}, \epsilon, \{B_x e_1, \ldots, B_x e_d\}, \|\cdot\|_q\right) \geq \mathcal{N}_\infty\left(\mathcal{F}, \epsilon, N, \|\cdot\|_q\right)$. To do this, let $\hat{\mathcal{F}}$ be a set of size $\mathcal{N}_\infty\left(\mathcal{F}, \epsilon, \{B_x e_1, \ldots, B_x e_d\}, \|\cdot\|_q\right)$ such that it covers $\mathcal{F}$ on the set $\{B_x e_1, \ldots, B_x e_d\}$ to size ϵ. We claim $\hat{\mathcal{F}}$ also covers $\mathcal{F}$ over any set of size N in our input space. Let $\hat{\mathcal{W}}$ refer to the matrices used in $\hat{\mathcal{F}}$. Let $W \in \mathcal{W}$ and let $\hat{W} \in \hat{\mathcal{W}}$ be the matrix we would choose to cover $\mathcal{W}$. Notice for any $\|x\|_1 \leq B_x$, we have

$$\left\|(W - \hat{W})x\right\|_q = \left\|\sum_{i=1}^{d}(W - \hat{W})x_i e_i\right\|_q \leq \sum_{i=1}^{d}|x_i|\left\|(W_i - \hat{W}_i)\right\|_q \leq B_x \max_{i\in[d]}\left(\left\|(W_i - \hat{W}_i)\right\|_q\right) =$$
$$\max_{i\in[d]}\left(\left\|(W - \hat{W})B_x e_i\right\|_q\right) \leq \epsilon$$

Thus for any set of $\{x_i\}_{i=1}^N$ where $\|x\|_1 \leq B_x$ we have

$$\sup_{j\in[N]}\left\|(W - \hat{W})x_j\right\|_q \leq \sup_{j\in[N]}\max_{i\in[d]}\left(\left\|(W - \hat{W})B_x e_i\right\|_q\right) = \max_{i\in[d]}\left(\left\|(W - \hat{W})B_x e_i\right\|_q\right) \leq \epsilon$$

which shows that it is also an upper bound and thus we have an equality.

A.3 Proof of Single Layer Bound (Theorem 2.4.1)

Before we start the analysis, for any vectors $v, u \in \mathbb{R}^d$, $\|v\|_1 \leq B_v$, notice the following inequality:

$$v^\top u \leq B_v \max_{j\in[d]}|e_j u| = \max_{j\in[d], s\in\{-1,1\}} s e_j u$$

We will also need this lemma that can be found in Edelman et al. 2022

Lemma A.3.1. *(Corollary A.7 in Edelman et al. 2022) For $\theta_1, \theta_2 \in \mathbb{R}^p$, we have*

$$\|\mathit{softmax}(\theta_1) - \mathit{softmax}(\theta_2)\|_1 \leq 2\|\theta_1 - \theta_2\|_\infty$$

Using the fist inequality above we can see we get:

$$\mathbb{E}\left[\sup_{w, W_c, W_v, W_{QK}}\sum_{i=1}^{m}\epsilon_i w^\top W_c^\top \sigma\left(W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right)\right] \leq$$
$$B_w \mathbb{E}\left[\sup_{s\in\{\pm1\}, j\in[d] W_c, W_v, W_{QK}} s\sum_{i=1}^{m}\epsilon_i e_j^\top W_c^\top \sigma\left(W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right)\right]$$

But now notice that $e_j^\top W_c^\top$ is also just a row vector. Thus we can use the inequality again to get:

$$B_w B_{W_c} \mathbb{E}\left[\sup_{s\in\{\pm 1\}, j\in[k], W_v, W_{QK}} s\sum_{i=1}^{m} \epsilon_i e_j^\top \sigma\left(W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right)\right] \leq$$

Since σ is applied elementwise, we can bring $e_j^\top$ inside of the function. Also, since $\sigma(0) = 0$, we can see that if we have W_v be the zero matrix, we have 0 in our function class. Therefore, we can get rid of the sign function by using the well known property of Rademacher complexities of:

$$Rad_m(\mathcal{F} \cup -\mathcal{F}, S) \leq 2Rad_m(\mathcal{F}, S)$$

This, along with the contraction inequality Ledoux and Talagrand [1991] allows us to upper bound the above by

$$2B_w B_{W_c} L_\sigma \mathbb{E}\left[\sup_{j\in[k], W_v, W_{QK}} \sum_{i=1}^{m} \epsilon_i e_j^\top W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right]$$

Continuing using the first inequality in this section again, we see

$$2B_w B_{W_c} L_\sigma \mathbb{E}\left[\sup_{j\in[k], W_v, W_{QK}} \sum_{i=1}^{m} \epsilon_i e_j^\top W_v^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right] \leq$$
$$2B_w B_{W_c} L_\sigma B_{W_v} \mathbb{E}\left[\sup_{s\in\{\pm 1\}, j\in[d], W_{QK}} \sum_{i=1}^{m} s\epsilon_i e_j^\top X_{(i)}^\top \text{softmax}\left(X_{(i)} W_{QK} x_{[CLS]}\right)\right]$$

Let us call the expectation above E. We will now use covering numbers and Dudley's integral to bound E to get our final generalization bound. First, we will use our covering number bound at scale $\epsilon' = \epsilon/2B_x^2$. Specifically, we will show that a set with a log size of $\ln(2d) + \frac{4B_x^4 C}{\epsilon^2}$ covers the function class in the expectation above. The $\ln(2d)$ comes from the fact that we will have to modify $\hat{\mathcal{W}}$ to have it work for us. We do this by using the covering function class

$$S = \{X \to s e_j^\top X^\top \text{softmax}\left(X \hat{W}_{QK} x_{[CLS]}\right) \mid s \in \{-1, 1\}, j \in [d], \hat{W}_{QK} \in \hat{\mathcal{W}}\}$$

Notice this allows for every $\hat{W}_{QK} \in \hat{\mathcal{W}}$, we have every combination of s and e_j.

Now, using the lemma A.3.1 and the linear algebra results of $\|Pv\| \leq \|P\|_{2,\infty} \|v\|_1$ and

$\|X\|_{2\to\infty} = \left\|X^\top\right\|_{2,\infty}$, we get:

$$\begin{aligned}
&\left\|se_j^\top X_{(i)}^\top \text{softmax}\left(X_{(i)}W_{QK}x_{[CLS]}\right) - se_j^\top X_{(i)}^\top \text{softmax}\left(X_{(i)}\hat{W}_{QK}x_{[CLS]}\right)\right\| \leq \\
&\left\|X_{(i)}^\top \text{softmax}\left(X_{(i)}W_{QK}x_{[CLS]}\right) - X_{(i)}^\top \text{softmax}\left(X_{(i)}\hat{W}_{QK}x_{[CLS]}\right)\right\| \leq \\
&\left\|X_{(i)}^\top\right\|_{2,\infty} \left\|\text{softmax}\left(X_{(i)}W_{QK}x_{[CLS]}\right) - \text{softmax}\left(X_{(i)}\hat{W}_{QK}x_{[CLS]}\right)\right\|_1 \leq \\
&2\left\|X_{(i)}^\top\right\|_{2,\infty} \left\|X_{(i)}W_{QK}x_{[CLS]} - X_{(i)}\hat{W}_{QK}x_{[CLS]}\right\|_\infty \leq \\
&2\left\|X_{(i)}^\top\right\|_{2,\infty}^2 \left\|W_{QK}x_{[CLS]} - \hat{W}_{QK}x_{[CLS]}\right\| \leq \\
&2B_X^2 \frac{\epsilon}{2B_X^2} = \epsilon
\end{aligned}$$

Therefore, we can see the log covering number of S is $\ln(2d) + \frac{4B_x^4 C}{\epsilon^2}$. Also, notice, due to the softmax in S, the largest value the function class can be is B_x. Then, using Dudley's integral we have a constant c such that

$$\begin{aligned}
&E/c \leq \inf_{\delta\geq 0} \delta + \int_\delta^{B_x} \sqrt{\frac{\ln(2d)}{m} + \frac{\frac{4B_x^4 C}{\epsilon^2}}{m}} d\epsilon \leq \\
&\inf_{\delta\geq 0} \delta + (B_x - \delta)\sqrt{\frac{\ln(2d)}{m}} + \int_\delta^{B_x} \sqrt{\frac{\frac{4B_x^4 C}{\epsilon^2}}{m}} d\epsilon \leq \\
&\inf_{\delta\geq 0} \delta + (B_x - \delta)\sqrt{\frac{\ln(2d)}{m}} + \frac{2B_x^2\sqrt{C}}{\sqrt{m}} \ln(B_x/\delta)
\end{aligned}$$

When $m > \ln(2d)$ standard analysis can find the minimum for δ is when

$$\delta = \frac{\frac{2B_x^2\sqrt{C}}{\sqrt{m}}}{1 - \sqrt{\frac{\ln(2d)}{m}}} = \frac{2B_x^2\sqrt{C}}{\sqrt{m} - \sqrt{\ln(2d)}}$$

Substituting this in and rearranging we get

$$\begin{aligned}
&\frac{\frac{2B_x^2\sqrt{C}}{\sqrt{m}}}{1 - \sqrt{\frac{\ln(2d)}{m}}} \left(1 - \sqrt{\frac{\ln(2d)}{m}}\right) + B_x\sqrt{\frac{\ln(2d)}{m}} + \frac{2B_x^2\sqrt{C}}{\sqrt{m}} \ln\left(\frac{B_x(\sqrt{m} - \sqrt{\ln(2d)})}{2B_x^2\sqrt{C}}\right) = \\
&\frac{2B_x^2\sqrt{C}}{\sqrt{m}} + B_x\sqrt{\frac{\ln(2d)}{m}} + \frac{2B_x^2\sqrt{C}}{\sqrt{m}} \ln\left(\frac{\sqrt{m} - \sqrt{\ln(2d)}}{2B_x\sqrt{C}}\right)
\end{aligned}$$

Thus, multiplying this by c and substituting this in for E gives us our desired result.

A.4 Corollaries of Theorem 2.4.1

Corollary A.4.0.1. *Let us have the requirements needed for Theorem 2.4.1 along with* $\|x\|_1 \leq B_x$ *and* $\|W_{QK}\|_{1,\infty} \leq B_{W_{QK}}$. *Let*

$$B = B_w B_{W_c} L_\sigma B_{W_v}$$

and let

$$\alpha = 2B_{W_{QK}}\sqrt{d\log(2d+1)}$$

Then we have our Transformer Rademacher complexity being less than

$$O\left(B\left(\frac{B_x^3\alpha}{\sqrt{m}}\left(1+\ln\left(\frac{\sqrt{m}-\sqrt{\ln(2d)}}{B_x^2\alpha}\right)\right)+B_x\sqrt{\frac{\ln(2d)}{m}}\right)\right)$$

Corollary A.4.0.2. *Let us have the requirements needed for Theorem 2.4.1 along with* $\|x\|_1 \leq B_x$ *and* $\|W_{QK}\|_{2,1} \leq B_{W_{QK}}$ *Let*

$$B = B_w B_{W_c} L_\sigma B_{W_v}$$

and let

$$\alpha = 2B_{W_{QK}}\sqrt{2\log(d)}$$

Then we have our Transformer Rademacher complexity being less than

$$\hat{O}\left(B\left(\frac{B_x^3\alpha}{\sqrt{m}}\left(1+\ln\left(\frac{\sqrt{m}-\sqrt{\ln(2d)}}{B_x^2\alpha}\right)\right)+B_x\sqrt{\frac{\ln(2d)}{m}}\right)\right)$$

Where the $\hat{O}$ *denotes normal* O *but with some logarithms not containing* T, d, k *being omitted from inside the formula.*

A.5 Proof of Multiple Layers Covering Number (Theorem 2.4.2)

We will first start with some useful lemmas stated in Edelman et al. [2022]. The proofs of these lemmas will not be reproduced for ease of reading.

Lemma A.5.1. *(Lemma A.8 in Edelman et al. 2022) For* $\epsilon, C_i, \beta_i \geq 0$, $i \in [n]$ *the solution to*

$$\min_{\epsilon_1,\ldots,\epsilon_n}\sum_{i=1}^{n}\frac{C_i}{\epsilon_i^2}$$

given

$$\sum_{i=1}^{n} \beta_i \epsilon_i = \epsilon$$

is $\frac{\gamma^3}{\epsilon^2}$ *where*

$$\gamma = \sum_{i=1}^{n} C_i^{1/3} \beta_i^{2/3}$$

and

$$\epsilon_i = \frac{\epsilon}{\gamma} \left(\frac{C_i}{\beta_i} \right)^{1/3}$$

The proof is by using Lagrange multipliers.

Lemma A.5.2. *(Lemma A.15 from Edelman et al. 2022) Suppose* $W^{1:i+1}$, $\hat{W}^{1:i+1}$ *satisfy our norm bounds. Then we have*

$$\begin{aligned}
&\left\| (g_{block}^{(i+1)}(X; W^{1:i+1}) - g_{block}^{(i+1)}(X; \hat{W}^{1:i+1}))^\top \right\|_{2,\infty} \leq \\
&\left\| (W_c^{(i)} - \hat{W}_c^{(i)})^\top \sigma \left(\Pi_{norm} \left(f(g_{block}^{(i)}(X; \hat{W}^{1:i}); \hat{W}^{(i)}) \right) \right)^\top \right\|_{2,\infty} + \\
&L_\sigma B_{c2} B_{v2} (1 + 4B_{QK2}) \left\| (g_{block}^{(i)}(X; W^{1:i}) - g_{block}^{(i)}(X; \hat{W}^{1:i}))^\top \right\|_{2,\infty} + \\
&2L_\sigma B_{c2} B_{v2} \left\| (W_{QK}^{(i)} - \hat{W}_{QK}^{(i)})^\top g_{block}^{(i)}(X; \hat{W}^{1:i})^\top \right\|_{2,\infty} + \\
&L_\sigma B_{c2} \left\| (W_v^{(i)} - \hat{W}_v^{(i)})^\top g_{block}^{(i)}(X; \hat{W}^{1:i})^\top \right\|_{2,\infty}
\end{aligned}$$

Lemma A.5.3. *(Lemma A.16 from Edelman et al. 2022) Given* $W^{1:i+1}$, $\hat{W}^{1:i+1}, w, \hat{w}$, *and* $g_{scalar}(X; W^{1:L+1}, w) = w^\top g_{block}^{(L+1)}(X; W^{1:L+1})_{[CLS]}$, *we have:*

$$\begin{aligned}
&\left| g_{scalar}(X; W^{1:L+1}, w) - g_{scalar}(X; \hat{W}^{1:L+1}, \hat{w}) \right| \leq \\
&\|w\| \left\| g_{block}^{(L+1)}(X; W^{1:L+1})_{[CLS]} - g_{block}^{(L+1)}(X; \hat{W}^{1:L+1})_{[CLS]} \right\| + \\
&\left\| (w - \hat{w})^\top g_{block}^{(L+1)}(X; \hat{W}^{1:L+1})_{[CLS]} \right\|
\end{aligned}$$

The main proof ideas behind the above two lemmas is to unroll them, then use some norm properties and the triangle inequality to split them up.

Now given these, we will prove the multiple layers covering number theorem. Suppose we have our linear covering bound described in the theorem statement. Let $X_1, \ldots, X_m$ be the inputs that is within our norm bounds. Let $\mathcal{W}_v^{(i)}, \mathcal{W}_c^{(i)}, \mathcal{W}_{QK}^{(i)}$ be the sets of all possible values for $W_v^{(i)}, W_c^{(i)}, W_{QK}^{(i)}$ respectively. Let $\hat{\mathcal{W}}_v^{(i)}$ cover the function class $\{x \to W_v^\top x \mid W_v \in \mathcal{W}_v^{(i)}, \|x\|_2 \leq 1\}$, let $\hat{\mathcal{W}}_c^{(i)}$ cover the function class $\{x \to W_c^\top x \mid W_c \in \mathcal{W}_c^{(i)}, \|x\|_2 \leq 1\}$ and let $\hat{\mathcal{W}}_{QK}^{(i)}$ cover the function class $\{x \to W_{QK} x \mid W_{QK} \in \mathcal{W}_{QK}^{(i)}, \|x\|_2 \leq 1\}$ except for $\hat{\mathcal{W}}_{QK}^{(1)}$,

which covers the same function class, but with $\|x\| \leq B_x$. Let all of these classes be covered with mT points and let $\epsilon_v^{(i)}$, $\epsilon_c^{(i)}$, $\epsilon_{QK}^{(i)}$ be the resolution for each cover. Also, let $\hat{\mathcal{W}}$ cover $\{x \to w^\top x \mid w \in \mathcal{W}, \|x\| \leq 1\}$ at resolution ϵ_w. The exact value of these resolutions will be shown at the end.

We will show that for any $\epsilon > 0$ and for any $W^{1:L+1}$ that satisfies our norm bounds that there exists a

$$\hat{W}^{1:L+1} \in \hat{\mathcal{W}}_c^{(1)} \otimes \hat{\mathcal{W}}_v^{(1)} \otimes \hat{\mathcal{W}}_{QK}^{(1)} \otimes \cdots \otimes \hat{\mathcal{W}}_c^{(L)} \otimes \hat{\mathcal{W}}_v^{(L)} \otimes \hat{\mathcal{W}}_{QK}^{(L)} \otimes \hat{\mathcal{W}}$$

such that

$$\left| g_{scalar}(X; W^{1:L+1}, w) - g_{scalar}(X; \hat{W}^{1:L+1}, \hat{w}) \right| \leq \epsilon$$

To start, we will use lemma A.5.3 to get

$$\begin{aligned}
&\left| g_{scalar}(X; W^{1:L+1}, w) - g_{scalar}(X; \hat{W}^{1:L+1}, \hat{w}) \right| \leq \\
&\|w\| \left\| g_{block}^{(L+1)}(X; W^{1:L+1})_{[CLS]} - g_{block}^{(L+1)}(X; \hat{W}^{1:L+1})_{[CLS]} \right\| + \\
&\left\| (w - \hat{w})^\top g_{block}^{(L+1)}(X; \hat{W}^{1:L+1})_{[CLS]} \right\| \leq \\
&\|w\| \left\| (g_{block}^{(L+1)}(X; W^{1:L+1}) - g_{block}^{(L+1)}(X; \hat{W}^{1:L+1}))^\top \right\|_{2,\infty} + \epsilon_w
\end{aligned}$$

Now, we use lemma A.5.2 to see

$$\begin{aligned}
&\left\| (g_{block}^{(L+1)}(X; W^{1:i+1}) - g_{block}^{(L+1)}(X; \hat{W}^{1:i+1}))^\top \right\|_{2,\infty} \leq \\
&\left\| (W_c^{(L)} - \hat{W}_c^{(L)})^\top \sigma \left(\Pi_{norm} \left(f(g_{block}^{(L)}(X; \hat{W}^{1:L}); \hat{W}^{(L)}) \right) \right)^\top \right\|_{2,\infty} + \\
&L_\sigma B_{c2} B_{v2} (1 + 4B_{QK2}) \left\| (g_{block}^{(L)}(X; W^{1:L}) - g_{block}^{(L)}(X; \hat{W}^{1:L}))^\top \right\|_{2,\infty} + \\
&2L_\sigma B_{c2} B_{v2} \left\| (W_{QK}^{(i)} - \hat{W}_{QK}^{(i)})^\top g_{block}^{(i)}(X; \hat{W}^{1:i})^\top \right\|_{2,\infty} + \\
&L_\sigma B_{c2} \left\| (W_v^{(L)} - \hat{W}_v^{(L)})^\top g_{block}^{(L)}(X; \hat{W}^{1:L})^\top \right\|_{2,\infty}
\end{aligned}$$

Notice that if $C^{(i)} \in \mathbb{R}^{d \times T}$, $i \in [m]$, $W \in \mathbb{R}^{k \times d}$

$$\max_{i \in [m]} \left\| (W - \hat{W}) C^{(i)} \right\|_{2,\infty} = \max_{i \in [m], t \in [T]} \left\| (W - \hat{W}) C_t^{(i)} \right\|$$

Therefore we can use our covering number bounds to bound the values in the $\|\cdot\|_{2,\infty}$.

Thus, we get

$$\begin{aligned}
&\left\|(g_{block}^{(L+1)}(X;W^{1:i+1}) - g_{block}^{(L+1)}(X;\hat{W}^{1:i+1}))^\top\right\|_{2,\infty} \leq \\
&\epsilon_c^{(L)} + L_\sigma B_{c2}\epsilon_v^{(L)} + 2L_\sigma B_{c2}B_{v2}\epsilon_{QK}^{(L)} + \\
&L_\sigma B_{c2}B_{v2}(1+4B_{QK2})\left\|(g_{block}^{(L)}(X;W^{1:L}) - g_{block}^{(L)}(X;\hat{W}^{1:L}))^\top\right\|_{2,\infty}
\end{aligned}$$

Now note how we can iteratively do this for $\left\|(g_{block}^{(L)}(X;W^{1:L}) - g_{block}^{(L)}(X;\hat{W}^{1:L}))^\top\right\|_{2,\infty}$ until we have gotten to the base case of $g_{block}^{(1)}(X;\hat{W}^{1:1})) = X$. Thus, if we let

$$\alpha_i = \prod_{j=i+1}^{L} L_\sigma B_{c2}B_{v2}(1+4B_{QK2})$$

we can see that

$$\begin{aligned}
&\max_{i\in m}\left|g_{scalar}(X_i;W^{1:L+1},w) - g_{scalar}(X_i;\hat{W}^{1:L+1},\hat{w})\right| \leq \\
&\epsilon_w + B_w\left(\sum_{i=2}^{L}\alpha_i(\epsilon_c^{(i)} + L_\sigma B_{c2}\epsilon_v^{(i)} + 2L_\sigma B_{c2}B_{v2}\epsilon_{QK}^{(i)})\right) + \\
&B_w\alpha_1\left(\epsilon_c^{(1)} + L_\sigma B_{c2}\epsilon_v^{(1)} + 2L_\sigma B_{c2}B_{v2}\epsilon_{QK}^{(1)}\right)
\end{aligned}$$

We left the first layer outside of the sum so we can recall $\epsilon_{QK}^{(1)}$ has a different input bound than the rest. Now, we can use lemma A.5.1 to get our desired sizes for our different ϵ's and get our covering number stated in the theorem.

A.6 Other Corollaries of Theorem 2.4.2

Corollary A.6.0.1. *Suppose we have the norm bounds required in lemma 2.3.3 for each $W_c^{(i)}, W_v^{(i)}, W_{QK}^{(i)}, w$ and let the maximum be B. Let B_x be the input bound. Suppose we also have the bounds needed for theorem 2.4.2. Let*

$$\begin{aligned}
\alpha_i &= \prod_{j=i+1}^{L} L_\sigma B_{c2}B_{v2}(1+4B_{QK2}) \\
\tau_i &= \alpha_i^{2/3} + (2\alpha_i L_\sigma B_{c2}B_{v2})^{2/3} + (\alpha_i L_\sigma B_{v2})^{2/3} \\
\gamma &= \left(dB^2B_x^2\ln(2k+1)\right)^{1/3}(2L_\sigma B_{c2}B_{v2}\alpha_1 B_w)^{2/3} + \\
&\left(dB^2\ln(2k+1)\right)^{1/3}\left(1+(B_w L_\sigma B_{v2})^{2/3}\right) \\
\eta &= \left(dB^2\ln(2k+1)\right)^{1/3}\left(B_w^{2/3}\sum_{i=2}^{L}\tau_i\right)
\end{aligned}$$

Then, the log covering number of g^{L+1}_{scalar} *is*

$$\frac{(\gamma+\eta)^3}{\epsilon^2}$$

Corollary A.6.0.2. *Suppose we have the norm bounds required in lemma 2.3.4 for each* $W_c^{(i)}, W_v^{(i)}, W_{QK}^{(i)}, w$ *and let the maximum be* B*. Let* B_x *be the input bound. Suppose we also have the bounds needed for theorem 2.4.2. Let*

$$\begin{aligned}
\alpha_i &= \prod_{j=i+1}^{L} L_\sigma B_{c2} B_{v2}(1+4B_{QK2}) \\
\tau_i &= \alpha_i^{2/3} + (2\alpha_i L_\sigma B_{c2} B_{v2})^{2/3} + (\alpha_i L_\sigma B_{v2})^{2/3} \\
\gamma &= \left(B^2 B_x^2 \ln(dk)\right)^{1/3} (2L_\sigma B_{c2} B_{v2} \alpha_1 B_w)^{2/3} + \\
&\left(B^2 \ln(dk)\right)^{1/3} \left(1 + (B_w L_\sigma B_{v2})^{2/3}\right) \\
\eta &= \left(B^2 \ln(dk)\right)^{1/3} \left(B_w^{2/3} \sum_{i=2}^{L} \tau_i\right)
\end{aligned}$$

Then, the log covering number of g^{L+1}_{scalar} *is*

$$\frac{(\gamma+\eta)^3}{\epsilon^2}$$

A.7 Proof Cross Entropy with Softmax is ℓ_∞ Lipshitz

We want to show that $L(\hat{y}, y) = \sum_{i=1}^{K} -y_i \log(\text{softmax}(x)_i)$ satisfies the following:

$$\forall \hat{y}_1, \hat{y}_2 \in \mathbb{R}^K \quad |L(\hat{y}_1, y) - L(\hat{y}_2, y)| \leq 2 \left\|\hat{y}_1 - \hat{y}_2\right\|_\infty$$

where y is a one-hot encoded vector where index i is hot. Then, notice by the mean value theorem and Holder's inequality, we have:

$$\exists x \in \mathbb{R}^K \quad |L(\hat{y}_1, y) - L(\hat{y}_2, y)| \leq \left\|\nabla L(x, y)(\hat{y}_1 - \hat{y}_2)\right\|_1 \leq \left\|\nabla L(x, y)\right\|_1 \left\|(\hat{y}_1 - \hat{y}_2)\right\|_\infty$$

Thus, if we can bound $\left\|\nabla L(z, y)\right\|_1$, we are done. Notice we can rewrite our loss as:

$$L(x, y) = \sum_{i=1}^{K} -y_i \log\left(\frac{e^{x_i}}{\sum_{t=1}^{K} e^{x_t}}\right)$$

Only one value in the sum survives since only the i^{th} value in y is set to 1 and the others are set to 0. Thus, for i and for $j \neq i$:

$$\frac{\partial L(x,y)}{\partial x_i} = -\frac{\sum_{t=1}^{K} e^{x_t}}{e^{x_i}} \frac{e^{x_i}\left(\sum_{t=1}^{K} e^{x_t}\right) - e^{x_i}e^{x_i}}{\left(\sum_{t=1}^{K} e^{x_t}\right)^2} = -\frac{\sum_{t\neq i}^{K} e^{x_t}}{\sum_{t=1}^{K} e^{x_t}}$$

$$\frac{\partial L(x,y)}{\partial x_j} = \frac{1}{\sum_{t=1}^{K} e^{x_t}} e^{x_j} = \frac{e^{x_j}}{\sum_{t=1}^{K} e^{x_t}}$$

Thus for any x, we have:

$$\|\nabla L(x,y)\|_1 = \frac{\sum_{j\neq i}^{K} e^{x_j}}{\sum_{j=1}^{K} e^{x_j}} + \sum_{j\neq i} \frac{e^{x_j}}{\sum_{t=1}^{K} e^{x_t}} = \frac{2\sum_{j\neq i}^{K} e^{x_j}}{\sum_{t=1}^{K} e^{x_t}} \leq 2$$

APPENDIX B

Appendix for Chapter 3

B.1 Notes

B.1.1 Simulation study

To investigate the optimality of K_T-lookahead decoding, we run a simple simulation study. The code to create these plots can be found on Github here. The methodology went as follows:

1. Create a probability distribution over an alphabet. We do this by creating a Markov chain with m nodes and our sequences are a L length paths along this chain.

   (a) We create this Markov chain by having a starting distribution and its transition probabilities for each node be Dirichlet($\alpha_1, \dots \alpha_n$) distributed where $\alpha_1 = \cdots = \alpha_n = \alpha$.

2. Once a graph is created, for values of $L \in \{2, 4, 6, 8\}$, $N, K \in \{1, 2, 4, 6, 8\}$, $N, K \leq L$, we find which L length path is optimal for the Markov chain (representing the optimal sequence) and then see if our K_T-lookahead algorithm finds the optimal case.

3. For $\alpha \in \{.1, .25, .5, .75, 1, 10\}$ and $m \in \{2, 4, 6, 8\}$, do the above steps for each (α, m) pair 200 times and group them based on (α, m).

4. For each grouping, calculate the average KL-divergence of the sequence distribution with respect to the uniform distribution and calculate the average amount of times K_T-lookahead was N-gram Hamming optimal for length L. This is what is shown in the plots.

We chose the maximum amount of nodes m and sequence length L to be 8 as there are 8^8 different sequences at the maximum (about 16 million). We show in Corollary 3.4.2.1 that

there is no polynomial-time optimal decoder, thus we resort to brute force to go through all the combinations. Even with the use of a GPU, this still takes about 11 hours since we try every different α, m, N, K, L combinations described above. We also sped up computation by using memoization, but due to the exponential nature of increasing nodes or sequence length, we would start to run into memory issues if we made the length or amount of nodes larger.

The CPU used to run the simulations was an Intel 9th generation i7 and the GPU was an NVIDIA Geforce GTX 1660 Ti. The code itself was written in Python. The only non-standard package used was Numba [Lam et al., 2015]. This package allows for Python code to be compiled, allowing for better wall runtime for our code. We also use its CUDA support to be able to interact with our GPU. The code for the simulations is on Github.

For ties, we were unable to come up with a setting that would never have ties in the K_T-lookahead arg max. These ties come from if there is a reordering of the maximum path such that each node still proceeds to the same next node, they just do so in a different order (e.g. 15717 would output the same probability as 17157). We also note that floating point multiplication is non-associative as well, which would also make there be no ties when there should be in some cases. The way we try to remedy this is by rounding $g(y)$ for every output y to 15 decimal places and then choosing the arg max over those. If K_T-lookahead outputted a path that was in the arg max, we considered it optimal.

We know that K_1-lookahead when $K = N = L$ should always produce the correct result. Thus, we thought it reasonable to see the impact of these ties by counting how many incorrect sequences were found by K_1-lookahead when $K = L = N$ on 200 trials of the simulation above. We found that K_1-lookahead was at most 3.5% unoptimal over all αs, number of nodes, and values of $K = N = L$. All figures in this chapter are accounting for ties as described in the previous paragraph.

In Figure B.1 we give the a set of plots that shows K_1-lookahead optimality over all parameters in the simulation.

From this, we can see that the larger N gets, generally the better our K_1-lookahead does. Since K_T-lookahead is choosing to walk along the path of maximum probability for each iteration, it makes sense that larger Ns would reward this as the N-gram Hamming loss gets closer and closer to the $0-1$ loss. Smaller Ns are more concerned with the marginal probabilities at each index, something that K_T-lookahead does not directly concern itself with.

For completeness, we give similar plots seen for K_1-lookahead, but for K_K-lookahead in Figures B.2, B.3, and B.4.

We can see that every empirical claim made with the K_1-lookahead plots can also be

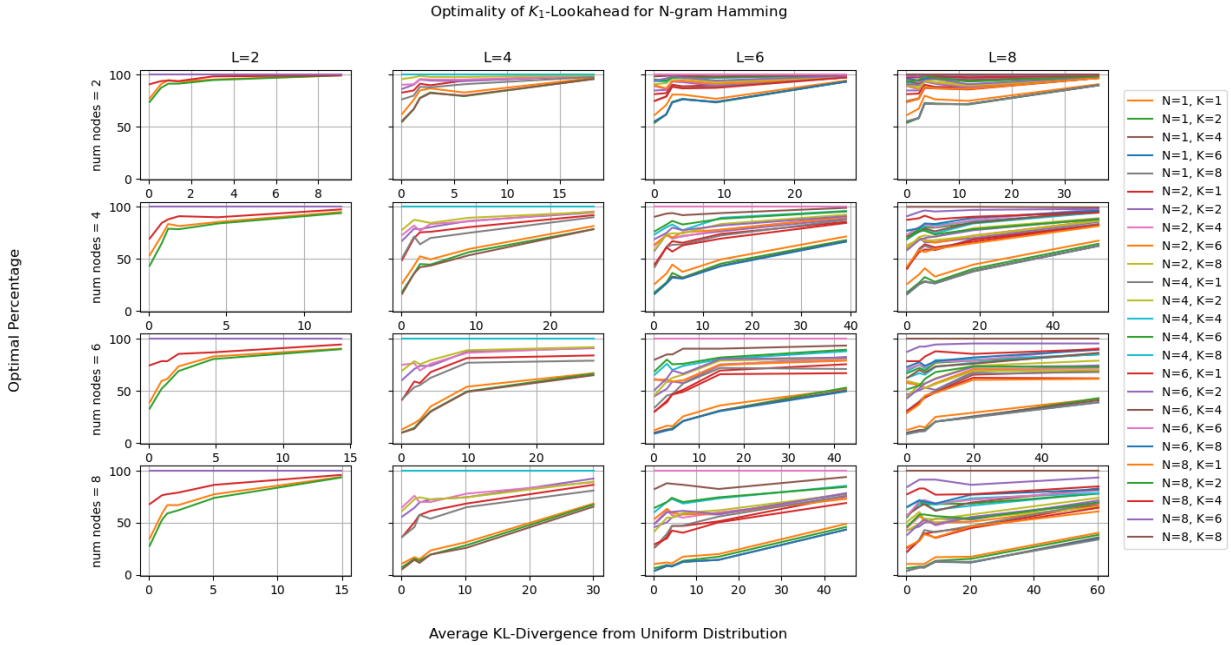


Figure B.1: A plot of all trials of K_1-lookahead decoding being optimal for N-gram Hamming on a Markov chain with "num nodes" amount of nodes. The amount of nodes increases as one looks down the rows and the sequence length increases as one goes down the columns. Each line represents a specific K_1-lookahead being compared to the optimal N-gram Hamming. The x and y axes denote the same that has been used in Figures 3.1 and 3.2.

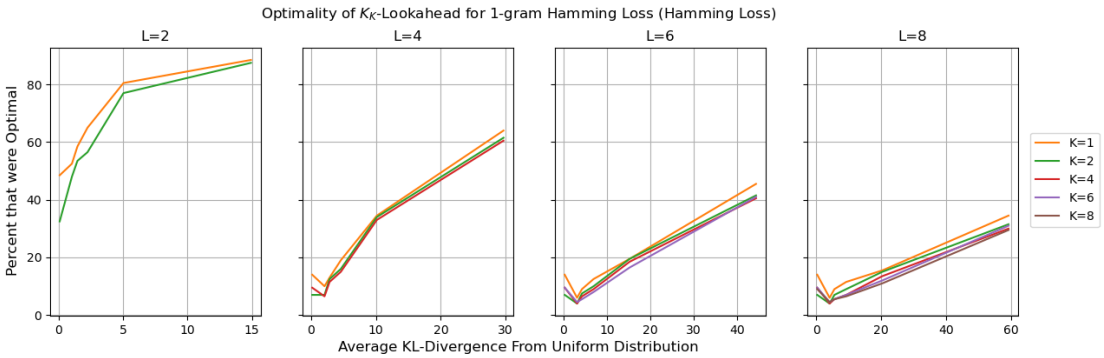


Figure B.2: A plot of the amount of trials K_K-lookahead was optimal for the 1-gram Hamming loss (the Hamming loss). There were 8 nodes in each Markov chain and the sequence length goes up by two as one moves right in the plots.

made here. Further, we see that K_1 and K_K-lookahead both output similar looking plots, with a possible slight edge towards K_1 in optimality. We do not make any claims that one is better than the other. For any T_1, T_2 where $T_1 < T_2$, K_{T_1} is going to require more compute time that K_{T_2}. Even though the probability distributions studied here are quite simple, a further investigation into if this extra computation is needed would be interesting given the

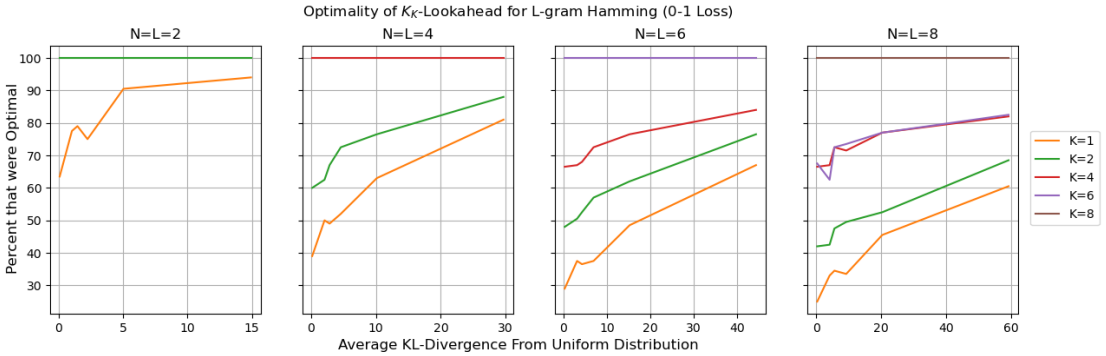


Figure B.3: A plot of the amount of trials K_K-lookahead was optimal for the L-gram Hamming loss (the $0-1$ loss). The same setup as Figure B.2 otherwise.

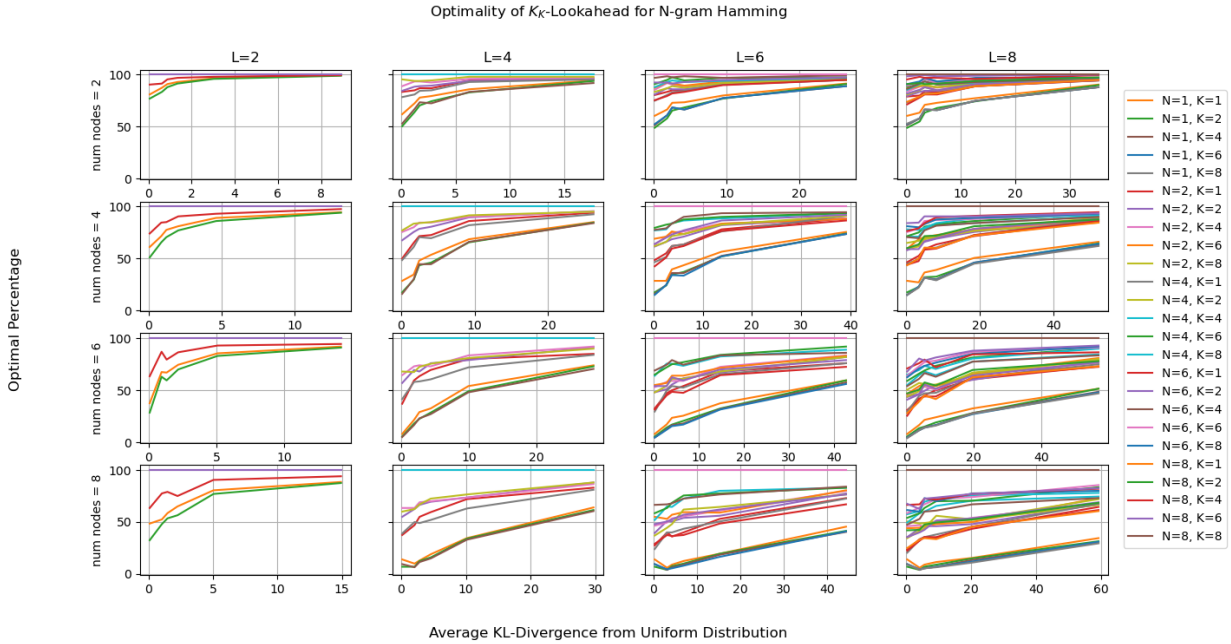


Figure B.4: A plot of all trials of K_K-lookahead decoding being optimal for N-gram Hamming on a Markov chain with "num nodes" amount of nodes. The amount of nodes increases as one looks down the rows and the sequence length increases as one goes down the columns. Each line represents a specific K_K-lookahead being compared to the optimal N-gram Hamming. The x and y axes denote the same that has been used in Figures B.2 and B.3.

similarity of optimality plots between K_1 and K_K-lookahead decoding.

B.2 Proofs and examples

B.2.1 Proof that temperature scaling is equivalent to our formulation

Let $\gamma = 1/T$ and let the set Z be our logits. Then, we have that

$$p(y_i \mid y_{[i-1]}) = \frac{e^{z_i}}{\sum_{z_j \in Z} e^{z_j}}$$

Then, we have

$$\frac{p(y_i \mid y_{[i-1]})^\gamma}{\sum_{v \in \mathcal{V}} p(v \mid y_{[i-1]})^\gamma} = \frac{\frac{e^{z_i/T}}{\left(\sum_{z_j \in Z} e^{z_j}\right)^{1/T}}}{\sum_{z_r \in Z} \frac{e^{z_r/T}}{\left(\sum_{z_j \in Z} e^{z_j}\right)^{1/T}}} =$$

$$\frac{\left(\frac{1}{\left(\sum_{z_j \in Z} e^{z_j}\right)^{1/T}}\right) e^{z_i/T}}{\left(\frac{1}{\left(\sum_{z_j \in Z} e^{z_j}\right)^{1/T}}\right) \sum_{z_r \in Z} e^{z_r/T}} = \frac{e^{z_i/T}}{\sum_{z_r \in Z} e^{z_r/T}}$$

This last value is how temperature scaling is implemented.

We do note this T is different than the T used in the rest of the chapter for K_T-lookahead. We use T here to represent the softmax temperature as it is the standard notation for it, but nowhere else in this chapter do we use it to represent temperature.

B.2.2 Proof of assertion in Assumption 3.3.1

Lemma B.2.1. *Suppose we have* $\forall i \in [L]$, $\forall y_{[i]} \in \mathcal{Y}_{[i]}$, *and* $\forall v \in \mathcal{V}$,

$$p^i_{ntp}(v \mid y_{[i]}) \to p^*(v \mid y_{[i]}).$$

Then $KL(p^* || p^i_{ntp}) \to 0$.

Proof. Below we begin by expanding the KL-divergence out and using expectation and log

properties to decompose it into a function of the conditional KL-Divergences.

$$KL(p^*||p^i_{ntp}) = \mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(\frac{p^i_{ntp}(y)}{p(y)}\right)\right] = \sum_{j=1}^{L}\mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right)\right] =$$
$$\sum_{j=1}^{L}\sum_{y\in\mathcal{Y}} -p^*(y)\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right) =$$
$$\sum_{j=1}^{L}\sum_{y\in\mathcal{Y}} -p^*(y_{[j-1]})p^*(y_j \mid y_{[j-1]})p^*(y_{j+1:} \mid y_{[j]})\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right) \qquad (\star)$$

Let us create, for each $i \in [L]$:

$$Y_{[i]} = \{y_{[i]} \mid y \in \mathcal{Y}\},$$
$$Y_{i+1:} = \{y_{i+1:} \mid y \in \mathcal{Y}\}.$$

We can see the inner sum then becomes:

$$\sum_{y_{[j]}\in\mathcal{Y}_{[j]}}\sum_{y_{j+1:}\in\mathcal{Y}_{j+1:}} -p^*(y_{[j-1]})p^*(y_j \mid y_{[j-1]})p^*(y_{j+1:} \mid y_{[j]})\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right) =$$
$$\sum_{y_{[j]}\in\mathcal{Y}_{[j]}} -p^*(y_{[j-1]})p^*(y_j \mid y_{[j-1]})\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right)\sum_{y_{j+1:}\in\mathcal{Y}_{j+1:}} p^*(y_{j+1:} \mid y_{[j]}).$$

From the definition of $\mathcal{Y}_{j+1:}$, we have that this last sum is 1. Thus, $(\star)$ becomes:

$$\sum_{j=1}^{L}\sum_{y_{[j]}\in\mathcal{Y}_{[j]}} -p^*(y_{[j-1]})p^*(y_j \mid y_{[j-1]})\log\left(\frac{p^i_{ntp}(y_j \mid y_{[j-1]})}{p(y_j \mid y_{[j-1]})}\right) =$$
$$\sum_{j=1}^{L}\sum_{y_{[j-1]}\in\mathcal{Y}_{[j-1]}}\sum_{v\in\mathcal{V}} -p^*(y_{[j-1]})p^*(v \mid y_{[j-1]})\log\left(\frac{p^i_{ntp}(v \mid y_{[j-1]})}{p(v \mid y_{[j-1]})}\right) =$$
$$\sum_{j=1}^{L}\sum_{y_{[j-1]}\in\mathcal{Y}_{[j-1]}} p^*(y_{[j-1]})\sum_{v\in\mathcal{V}} -p^*(v \mid y_{[j-1]})\log\left(\frac{p^i_{ntp}(v \mid y_{[j-1]})}{p(v \mid y_{[j-1]})}\right) =$$
$$\sum_{j=1}^{L}\sum_{y_{[j-1]}\in\mathcal{Y}_{[j-1]}} p^*(y_{[j-1]})\mathop{\mathbb{E}}_{v\sim p^*(\cdot|y_{[j-1]})}\left[-\log\left(\frac{p^i_{ntp}(v \mid y_{[j-1]})}{p(v \mid y_{[j-1]})}\right)\right].$$

By assumption, we have that

$$\mathop{\mathbb{E}}_{v\sim p^*(\cdot|y_{[j-1]})}\left[-\log\left(\frac{p^i_{ntp}(v \mid y_{[j-1]})}{p(v \mid y_{[j-1]})}\right)\right] \to 0$$

for each one. Thus, since there are a finite number of terms, standard arguments show that

that the entire function will limit to 0. □

B.2.3 Proof of Proposition 1

Proof. We will show that the limit of the difference of the expected risks is 0.

$$\mathop{\mathbb{E}}_{x\sim p_x, y\sim p^*|x, \hat{y}\sim p_{\mathcal{D}(p^i_{ntp})|x}}[\ell(\hat{y},y)] - \mathop{\mathbb{E}}_{x\sim p_x, y\sim p^*|x, \hat{y}\sim p_{\mathcal{D}(p^*_{ntp})|x}}[\ell(\hat{y},y)] =$$
$$\mathop{\mathbb{E}}_{x\sim p_x, y\sim p^*|x}\left[\sum_{\hat{y}\in\mathcal{Y}}\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\ell(\hat{y},y)\right] =$$
$$\mathop{\mathbb{E}}_{x\sim p_x}\left[\sum_{\hat{y}\in\mathcal{Y}}\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right] =$$
$$\sum_{\hat{y}\in\mathcal{Y}}\mathop{\mathbb{E}}_{x\sim p_x}\left[\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right].$$

Now, by assumption we have

$$p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y}) \to 0$$

and notice that $\left|\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathbb{E}_{y\sim p^*|x}[\ell(\hat{y},y)]\right| \leq M$. Thus, by the dominated convergence theorem, we have

$$\lim_{i\to\infty}\mathop{\mathbb{E}}_{x\sim p_x}\left[\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right] =$$
$$\mathop{\mathbb{E}}_{x\sim p_x}\left[\lim_{i\to\infty}\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right] = 0.$$

Thus, since $|\mathcal{Y}| < \infty$, we have

$$\lim_{i\to\infty}\sum_{\hat{y}\in\mathcal{Y}}\mathop{\mathbb{E}}_{x\sim p_x}\left[\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right] =$$
$$\sum_{\hat{y}\in\mathcal{Y}}\lim_{i\to\infty}\mathop{\mathbb{E}}_{x\sim p_x}\left[\left(p_{\mathcal{D}(p^i_{ntp})|x}(\hat{y}) - p_{\mathcal{D}(p^*_{ntp})|x}(\hat{y})\right)\mathop{\mathbb{E}}_{y\sim p^*|x}[\ell(\hat{y},y)]\right] = 0,$$

which is what we needed to show □

B.2.4 Proof of Lemma 3.4.1

Proof. Let $\hat{y}$ be the output of our algorithm. By linearity of expectation, we have

$$\mathbb{E}\left[\sum_{i=1}^{L-N+1} 1_{\{\hat{y}_{i:i+N-1} \neq y_{i:i+N-1}\}}\right] = \sum_{i=1}^{L-N+1} \mathbb{E}\left[1_{\{\hat{y}_{i:i+N-1} \neq y_{i:i+N-1}\}}\right]$$

We can see that

$$\mathbb{E}\left[1_{\{\hat{y}_{i:i+N-1} \neq y_{i:i+N-1}\}}\right] = 1 - p(\hat{y}_{i:i+N-1})$$

Therefore, we have

$$\mathbb{E}\left[\sum_{i=1}^{L-N+1} 1_{\{\hat{y}_{i:i+N-1} \neq y_{i:i+N-1}\}}\right] = \sum_{i=1}^{L-N+1} 1 - p(\hat{y}_{i:i+N-1}) =$$
$$L - N + 1 - \sum_{i=1}^{L-N+1} p(\hat{y}_{i:i+N-1}) = L - N + 1 - g(\hat{y})$$

Therefore, maximizing $g(y)$ will minimize our expected risk. □

B.2.5 Proof of Theorem 3.4.2

Proof. We have that

$$\forall y \in \mathcal{Y}, \quad p(y) = p(y_1)p(y_2|y_1)\dots p(y_L|y_{[L-1]}).$$

We can think of this as a path $y_1 \to y_2 \to \cdots \to y_L$. We can combine all these paths to make a directed tree. Let each node have the weight of the conditional distribution at that node. Thus, if we take the product of any path $y_1 \to y_2 \to \cdots \to y_L$, we get $p(y)$.

Suppose p is defined as in the Theorem 3.4.2 and suppose we do not query $|\mathcal{V}|^L - 1$ conditional probability values. We note that $|\mathcal{V}|^L - 1 = \frac{|\mathcal{V}|-1}{|\mathcal{V}|}\sum_{j=1}^{L}|\mathcal{V}|^j$. We know that the j^{th} level in our tree has $|\mathcal{V}|^j$ nodes. Therefore, on at least one level, the ratio of nodes we have queried is less than $\frac{|\mathcal{V}|-1}{|\mathcal{V}|}$. Thus, by the pigeonhole principle, there must exist two nodes that have the same path up until that point, $v|y_{[j-1]}$, $u|y_{[j-1]}$, that have not been looked at. Therefore, the algorithm is unable to know the exact probability of any descendants of these two paths. If either of these nodes have a weight more than $1/|\mathcal{V}|$, say without loss of generality it is $v|y_{[j-1]}$, then $g(y_{[j-1]} + v + \dots)$ would be larger than any path $\mathcal{D}$ has found so far. Therefore, the algorithm can not be sure either answer is optimal and must query more. Thus, since the algorithm was arbitrary, on this distribution any algorithm runtime will be at least $C(|\mathcal{V}|^L - 1)$. □

B.2.6 Proof of Lemma 3.4.3

Proof. Let $x \in \mathcal{X}$, $k \in \mathbb{N}$ and let $p^i|x \to p^*|x$. Then, let

$$\epsilon_i = \max\left\{\left|p^i(y_{cK+1}, \ldots, y_{cK+K} \mid y_{[cK]}, x) - p^*(y_{cK+1}, \ldots, y_{cK+K} \mid y_{[cK]}, x)\right| \mid c \in \mathbb{Z}_+, y \in \mathcal{Y}\right\}$$

Notice, in order for $p^i|x \to p^*|x$, we need $\epsilon_i \to 0$. Thus, since there are only a finite amount of marginal and conditional values our decoder can look at, and since we know there are no ties, there will be some j such that $\forall i > j$ the arg max for the conditional distributions of both p^i and p^* will match. Therefore, we meet the assumption needed to use Prop 1. □

B.2.7 Proof of Proposition 2

Proof.

- $p_x^*(C) = 1 \implies$ **K_T-lookahead optimality**: By the defintion of C, we know the K_T-lookahead outputs maximize $\sum_{i=1}^{L-N+1} p^*(y_{i:i+N-1} \mid x)$ except a set of measure 0 over X. From Lemma 3.4.1, we can see that this is the optimal output.

- **K_T-lookahead optimality** $\implies$ $p_x^*(C) = 1$: We will prove the contraposition. Suppose $p_x^*(C) < 1$. Then, there exists a set $L \in \mathcal{X}$ of measure > 0 where for each $x \in L$:

$$\arg\max_y \sum_{i=1}^{L-N+1} p^*(y_{i:i+N-1} \mid x) = y^\dagger,$$

but

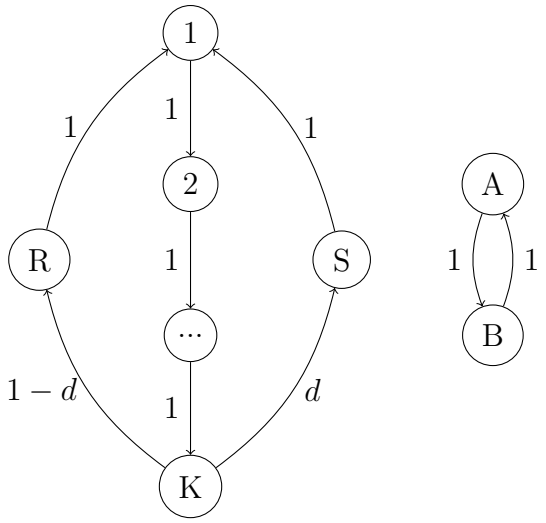
$$y^\dagger \neq \hat{y},$$

where $\hat{y}$ what our K_T-lookahead decoding algorithm outputs. Since we know $y^\dagger$ is optimal, K-lookahead decoding will be unoptimal.

□

B.2.8 Example of a Markov chain that is not K_T-lookahead optimal

Let $K, L, N, T \in \mathbb{N}$ such that $T \leq K < L$ and $N \leq L$ so that this set up makes sense in the context of this chapter. Let us have the following Markov chain:



for some $\frac{1}{2} < d < 1$. Then, note how K_T-lookahead will repeat $123 \dots KS$ to length L. Let $s = s_1 s_2 \dots s_L$ be the output of K_T-lookahead and let $a = a_1 \dots a_L$ be the sequence $ABAB \dots$ to size L. We can see that that amount of a_i's such that $a_i = S$ will be $\left\lfloor \frac{L}{K+1} \right\rfloor$. We can see that, since $K < L$, we have $1 \leq \left\lfloor \frac{L}{K+1} \right\rfloor$. Now, let c be the amount of N-grams in s that contain an S. Since $1 \leq \left\lfloor \frac{L}{K+1} \right\rfloor$, we have $1 \leq c$. Let

$$0 < \epsilon < \frac{1}{4\left(L - N + 1 - \frac{1-d}{2}c\right)}.$$

We can see that our upper bound is always greater than 0 as $c \leq L - N + 1$ and $d < 1$, therefore this ϵ is valid for any possible values of K, T, N, L and d that work in this setting. Let our initial distribution of the Markov chain, π, be such that $\mathbb{P}_\pi(1) = .50 + \epsilon$ and $\mathbb{P}_\pi(A) = .50 - \epsilon$. If $s_i \dots s_{i+N-1}$ is an N-gram of s that contains S, we have that

$$\mathbb{P}(s_i \dots s_{i+N-1}) \leq d\left(\frac{1}{2} + \epsilon\right)$$

because there is at least one S in it (and possibly more if N is large enough). If it does not contain an S, then we can see that

$$\mathbb{P}(s_i \dots s_{i+N-1}) = \frac{1}{2} + \epsilon.$$

Therefore, we have the following:

$$\sum_{n=1}^{L-N+1} \mathbb{P}(s_n \dots s_{n+N-1}) \leq cd\left(\frac{1}{2} + \epsilon\right) + \sum_{n=1}^{L-N+1-c}\left(\frac{1}{2} + \epsilon\right) =$$
$$cd\left(\frac{1}{2} + \epsilon\right) + (L - N + 1 - c)\left(\frac{1}{2} + \epsilon\right).$$

We can also see that for the sequence a:

$$\sum_{n=1}^{L-N+1} \mathbb{P}\left(a_n \dots a_{n+N-1}\right) = (L-N+1)\left(\frac{1}{2}-\epsilon\right).$$

In order for K_T-lookahead to not be optimal, we need

$$(L-N+1)\left(\frac{1}{2}-\epsilon\right) > cd\left(\frac{1}{2}+\epsilon\right) + (L-N+1-c)\left(\frac{1}{2}+\epsilon\right).$$

From this, below we manipulate algebra to arrive that ϵ must be smaller than its upper bound.

$$\begin{aligned}
&(L-N+1)\left(\frac{1}{2}-\epsilon\right) > cd\left(\frac{1}{2}+\epsilon\right) + (L-N+1-c)\left(\frac{1}{2}+\epsilon\right) \implies \\
&\frac{1}{2}(L-N+1) - \epsilon(L-N+1) > \frac{1}{2}\left(L-N+1-(1-d)c\right) + \epsilon\left(L-N+1-(1-d)c\right) \implies \\
&\frac{1}{2}\left(L-N+1-L+N-1+(1-d)c\right) > \epsilon\left(L-N+1+L-N+1-(1-d)c\right) \implies \\
&\epsilon < \frac{1}{2}\frac{(1-d)c}{2\left(L-N+1-\frac{1-d}{2}c\right)} = \frac{(1-d)c}{4\left(L-N+1-\frac{1-d}{2}c\right)}
\end{aligned}$$

Therefore, we have that a is a more optimal sequence for this Markov chain/initial distribution pair than the output of K_T-lookahead, which is what we wanted to show.

B.2.9 Proof of Proposition 3

Proof. Let $N < L$. Then, let us have the following probability distribution:

$$p(0\dots0) = .28, \quad p(\underbrace{0\dots0}_{K_2-1 \text{ indices}}\ 10\dots0) = .12, \quad p(20\dots0) = .23, \quad p(11\dots1) = .37.$$

One can verify that the $K_{1_{T_1}}$-lookahead decoder would output $0\dots0$ and the $K_{2_{T_2}}$-lookahead decoder would output $1\dots1$. One can also verify that for any N-gram starting at position $c > 1$,

$$\underbrace{0\dots0}_{N \text{ indices}} = \arg\max_{y_{c:c+N-1}} p(y_{c:c+N-1})$$

since $p(0_j\dots0_{j+N-1}) \geq .28+.23 = .51$ where $j > 1$. From this, by calculating the conditional and marginal distributions for the first N-gram, one can see that $\arg\max_{y\in\mathcal{Y}}\{g(y)\} = 0\dots0$. Therefore, the $K_{1_{T_1}}$-lookahead decoder is optimal for the n-gram Hamming loss, while the $K_{2_{T_2}}$-lookahead decoder is not.

For $N = L$, we have the $0-1$ loss, whose optimal output is the max probability sequence.

let us have the following probability distribution:

$$p(0\dots0) = .408, \quad p(\underbrace{0\dots0}_{K_2-1 \text{ indices}} 11\dots1) = .102,$$
$$p(\underbrace{1\dots1}_{K_2 \text{ indices}} 00\dots0) = .2401, \quad p(11\dots1) = .2499.$$

Here, we can see that the $K_{1_{T_1}}$-lookahead decoder will output $0\dots0$, however, since the max marginal for the first K_2 is $1\dots1$, the $K_{2_{T_2}}$-lookahead decoder will not output $0\dots0$.

Thus, we have covered all cases and have shown what was need. □

B.2.10 Proof of Proposition 4 and monotonicity result

Proof. Now, let N, K, L, T be as stated in Proposition 4. We will constructively create a two counterexamples, one for when $N < L$ and another for when $N = L$. Let our alphabet be $\{0, 1, 2\}$. For the $N < L$ case, we have the following probability distribution:

$$p(0\dots0) = .27675, \quad p(10\dots0) = .25, \quad p(\underbrace{0\dots0}_{K \text{ indices}} 20\dots0) = .03075,$$
$$p(\underbrace{0\dots0}_{T \text{ indices}} 1\dots1) = .2925, \quad p(\underbrace{0\dots0}_{T \text{ indices}} 20\dots0) = .15.$$

It is easy to see that both K_T and K_{T+1} will both choose $0\dots0$ for their first T and $T+1$ values respectively. From this, we can see that this locks in $T+1$ into choosing rather $0\dots0$ or $\underbrace{0\dots0}_{T \text{ indices}} 20\dots0$, from which one can see it will choose $0\dots0$ by following the algorithm. For K_T, it sees the following for its second iteration:

$$p(\underbrace{1\dots1}_{K \text{ indices}} \mid \underbrace{0\dots0}_{T \text{ indices}}) = .39, \quad p(\underbrace{0\dots0}_{K-T \text{ indices}} \underbrace{20\dots0}_{T \text{ indices}} \mid \underbrace{0\dots0}_{T \text{ indices}}) = .041,$$
$$p(\underbrace{0\dots0}_{K \text{ indices}} \mid \underbrace{0\dots0}_{T \text{ indices}}) = .369, \quad p(\underbrace{20\dots0}_{K \text{ indices}} \mid \underbrace{0\dots0}_{T \text{ indices}}) = .2.$$

From this, we can see that it will choose $\underbrace{1\dots1}_{T \text{ indices}}$ and then be locked into the sequence $\underbrace{0\dots0}_{T \text{ indices}} 1\dots1$. Now that we know both of the outputs of K_T and K_{T+1}, we need to show that $0\dots0$ is optimal. Notice that for any N-gram starting at position $c > 1$:

$$p(0_c\dots0_{c+N-1}) \geq .27675 + .25 = .52675$$

and there are no ties in the arg max. Let us then have a sequence y. Let S_y be all N-grams of y that contain a non-0 index. Notice that:

$$\forall y_{j:j+N-1} \in S_{y_{1:}} \quad p(y_{j:j+N-1}) < p(0_j \dots 0_{j+N-1}).$$

Thus, for every starting index greater than 1, our sequence would be better off it was only 0s. Therefore, we only need to show the same for index 1. By calculating the marginal and conditional distributions, it can be seen that, for every N, $p(0_1 \dots 0_N) > p(y_{1:N-1}) - .02$ where $y_{1:N-1}$ is any N-gram that is not all zeros. Therefore, since for any $c > 1$, $p(0_c \dots 0_{c+N-1}) > .5 + .02$, we know that

$$g(0 \dots 0) = \sum_{i=1}^{L-N+1} p(0_i \dots 0_{i+N-1}) > \sum_{i=1}^{L-N+1} p(y_i \dots y_{i+N-1}) = g(y)$$

for any other sequence y. Thus $0 \dots 0$ is optimal.

For $N = L$, let our alphabet be $\{0, 1, 2\}$. Now, we will define marginal and conditional probabilities for the first K indices for the probability distribution:

$$p(\underbrace{0 \dots 0}_{T \text{indices}}) = 1, \quad p(\underbrace{0 \dots 0}_{K-T \text{ indices}} \mid \underbrace{0 \dots 0}_{T \text{indices}}) = .51, \quad p(\underbrace{1 \dots 1}_{K-T \text{ indices}} \mid \underbrace{0 \dots 0}_{T \text{indices}}) = .49.$$

From this, we can see that K_{T+1} will choose $\underbrace{0 \dots 0}_{T+1 \text{ indices}}$ for the first round but K_T only chooses $\underbrace{0 \dots 0}_{T \text{ indices}}$. The goal now is to adversarially create the rest of the sequence probabilities so that K_T and K_{T+1} diverge and K_{T+1} is optimal. Let us now give the full probability distribution:

$$p(\underbrace{0 \dots 0}_{K \text{ indices}} 2 \dots 2) = .051, \quad p(0 \dots 0) = .459,$$
$$p(\underbrace{0 \dots 0}_{T \text{ indices}} 1 \dots 1) = .2499, \quad p(\underbrace{0 \dots 0}_{T \text{ indices}} \underbrace{1 \dots 1}_{K \text{ indices}} 0 \dots 0) = .2401.$$

We note that since $K < L - T \implies K + T < L$, the last two sequences above are distinct (i.e., there is at least one 0 at the end of the last sequence). Notice how we have created two paths that diverge at the $T + 1$ spot depending on if the $T + 1$ spot is a 0 or a 1. On the second iteration, K_T will see that $p(\underbrace{1 \dots 1}_{K \text{ indices}} \mid \underbrace{0 \dots 0}_{T \text{ indices}}) = .49$, while any other choices would have less probability than that, thus we have that K_T will choose 1 at the $T + 1$ spot. Since K_{T+1} already chose a 0 at that spot, their paths have split. Specifically, we can see that K_T

will choose $\underbrace{0\dots0}_{T \text{ indices}}\,1\dots1 = \hat{y}$ and K_{T+1} will choose $0\dots0 = y^{\dagger}$. Since $N = L$, we know the optimal sequence is the one with the most probability, which is $0\dots0$, which shows what we needed. □

For the monotonicity result, let $K \in \{2, 3\dots, L\}$, $N = L$, $T_1, T_2 \in [K]$ such that $T_1 < T_2$. Suppose also $K \geq L - T_1$. K_{T_1} and K_{T_2} are looking over the same K tokens in the first iteration, thus their first T_1 values will be the same. Then, since $K \geq L - T_1$, we know K_{T_1} will choose the optimal rest of the tokens since it looks over every possibility left. Therefore, we only need to know if its first T_1 tokens were optimal. Since K_{T_2} is optimal, and they share the same first T_1 tokens, we then know that K_{T_1} is optimal.

B.2.11 Proof of Proposition 5

Proof. Let p be our probability distribution over $\mathcal{X} \times \mathcal{Y}$ and let $\mathcal{D}$ be our decoding algorithm. By lemma 3.4.1, given an input x, the optimal output is $\arg\max_y g(y|x)$. Notice:

$$\mathop{\mathbb{E}}_{y\sim p|x, \hat{y}\sim p_{\mathcal{D}(p_{ntp})|x}} \left[\sum_{i=1}^{L-N+1} 1_{\{y_{i:i+N-1} \neq \hat{y}_{i:i+N-1}\}}\right] =$$
$$\sum_{\hat{y}\in\mathcal{Y}} p_{\mathcal{D}(p_{ntp})|x}(\hat{y}) \mathop{\mathbb{E}}_{y\sim p|x} \left[\sum_{i=1}^{L-N+1} 1_{\{y_{i:i+N-1} \neq \hat{y}_{i:i+N-1}\}}\right].$$

We know that $\sum_{\hat{y}\in\mathcal{Y}} p_{\mathcal{D}(p_{ntp})|x}(\hat{y}) = 1$. Therefore, in order to minimize our total sum, we need all the mass of $p_{\mathcal{D}(p_{ntp})|x}(\hat{y})$ to be on values of $\hat{y}$ which minimize our expected risk. Since this was for an arbitrary $x \in \mathcal{X}$, we have shown what was needed. □

B.2.12 Random sampling and temperature-scaled random sampling meet the assumption needed for Proposition 1

Let us look at one particular y. Let $p_{RS(p|x)}(\cdot)$ be the probability distribution of random sampling decoder using p as a next-token predictor given an input x and $p_{TSRS(p|x,\gamma)}(\cdot)$ be the same for temperature scaled random sampling with hyperparameter γ. By defintion, we have for every $x \in \mathcal{X}$ and $y \in \mathcal{Y}$:

$$p_{RS(p|x)}(y) = \prod_{i=1}^{L} p(y_i \mid y_{[i-1]}, x)$$
$$p_{TSRS(p|x,\gamma)}(y) = \prod_{i=1}^{L} \frac{p(y_i \mid y_{[i-1]}, x)^{\gamma}}{\sum_{v\in\mathcal{V}} p(v \mid y_{[i-1]}, x)^{\gamma}}.$$

We can see that each of these are continuous in $p(\cdot \mid \cdot)$ so long as $\gamma \neq \infty$. Thus, as $p^i \to p^*$, we have that

$$p_{RS(p^i|x)}(y) \to p_{RS(p^*|x)}(y)$$

and

$$p_{TSRS(p^i|x,\gamma)}(y) \to p_{TSRS(p^*|x,\gamma)}(y).$$

If $\gamma = \infty$, then this becomes greedy decoding, which we show in Lemma 3.4.3 meets the assumption needed as well.

B.2.13 Proof of Proposition 6

Proof. By the probability chain rule, we can see that random sampling from $p^i_{ntp}(\cdot \mid \cdot)$ and then concatenating has the same distribution as sampling from p^i itself. Therefore, we will work with p^i for the rest of the proof without regard for next-token prediction. Given that $H(p)$ is the entropy of a probability distribution p, we have

$$\begin{aligned}
&CE(p^i, p^*) = \\
&\mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(p^i(y)\right)\right] = \mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(p^i(y)\right)\right] + \mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(p^*(y)\right)\right] - \mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(p^*(y)\right)\right] = \\
&\mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(\frac{p^i(y)}{p^*(y)}\right)\right] + \mathop{\mathbb{E}}_{y\sim p^*}\left[-\log\left(p^*(y)\right)\right] = KL(p^*||p^i) + H(p^*).
\end{aligned}$$

By Assumption 3.3.1 we have that $p^i \to p^*$ in KL-Divergence. Since KL-Divergence is also a metric, we have that $CE(p^i, p^*) \geq H(p^*)$. Thus, we can see that $\lim_{i\to\infty} CE(p^i, p^*) = H(p^*)$, which shows we obtain the minimum value we can and therefore have consistency. □

B.2.14 Proof of Proposition 7

Proof. We know that $p^i \to p^*$ in KL-divergence. Further, by Appendix B.2.12, we can see that for all $y \in \mathcal{Y}$ and $x \in \mathcal{X}$, $p_{RS(p^i_{ntp}|x)}(y) \to p_{RS(p^*_{ntp}|x)}(y)$ and $p_{TSRS(p^i_{ntp}|x,\gamma)}(y) \to p_{TSRS(p^*_{ntp}|x,\gamma)}(y)$. Thus, if we can show that $p_{TSRS(p^*_{ntp}|x,\gamma)} \neq p_{RS(p^*_{ntp}|x)}$, then we are done. Let p be the limit for random sampling and let p^γ be the limit for temperature scaled random sampling for temperature parameter γ.

Let $\gamma \neq 1$. In section 3.5.2 we know that random sampling is optimal. In Appendix B.2.13 we show the well known fact that cross entropy is the sum of the entropy of the true distribution plus the KL-Divergence of the two distributions. The KL-divergence has a unique minimum at the true distribution. Thus, we will show that $KL(p||p^\gamma) = 0$ if and only if p is uniform or deterministic. We begin by using the same analysis done in Appendix

B.2.2 to break the KL-Divergence up into a function of the conditional KL-Divergences.

$$KL(p||p^\gamma) = \mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\frac{p^\gamma(y)}{p(y)}\right)\right] = \sum_{i=1}^{L}\mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\frac{\frac{p(y_i|y_{[i-1]})^\gamma}{\sum_{v\in\mathcal{V}}p(v|y_{[i-1]})^\gamma}}{p(y\mid y_{[i-1]})}\right)\right] =$$

$$\sum_{i=1}^{L}\mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right)\right] = \sum_{i=1}^{L}\sum_{y\in\mathcal{Y}} -p(y)\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right) =$$

$$\sum_{i=1}^{L}\sum_{y\in\mathcal{Y}} -p(y_{[i-1]})p(y_i\mid y_{[i-1]})p(y_{[i+1:]}\mid y_{[i]})\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right) =$$

$$\sum_{i=1}^{L}\sum_{y_{[i]}\in\mathcal{Y}_{[i]}} -p(y_{[i-1]})p(y_i\mid y_{[i-1]})\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right) =$$

$$\sum_{i=1}^{L}\sum_{y_{[i-1]}\in\mathcal{Y}_{[i-1]}} p(y_{[i-1]})\sum_{y_i\in\mathcal{V}} -p(y_i\mid y_{[i-1]})\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right) =$$

$$\sum_{i=1}^{L}\sum_{y_{[i-1]}\in\mathcal{Y}_{[i-1]}} p(y_{[i-1]})\mathop{\mathbb{E}}_{y_i\sim p(\cdot|y_{[i-1]})}\left[-\log\left(\frac{p(y_i\mid y_{[i-1]})^{\gamma-1}}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma}\right)\right] =$$

$$\sum_{i=1}^{L}\sum_{y_{[i-1]}\in\mathcal{Y}_{[i-1]}} p(y_{[i-1]})\mathop{\mathbb{E}}_{y_i\sim p(\cdot|y_{[i-1]})}\left[-\log\left(\frac{\frac{p(y_i|y_{[i-1]})^\gamma}{\sum_{v\in\mathcal{V}}p(v|y_{[i-1]})^\gamma}}{p(y_i\mid y_{[i-1]})}\right)\right].$$

Thus, we can see that $KL(p||p^\gamma)$ is a function of the KL-divergence of the conditional probability distributions. Since we need $KL(p||p^\gamma) = 0$, this would then make us need each conditional KL-divergence also need to be 0. Thus, we require for every $y_{[i-1]} \in \mathcal{Y}_{[i-1]}$ that has non-zero probability

$$\forall v \in \mathcal{V} \quad \frac{p(v\mid y_{[i-1]})^\gamma}{\sum_{v\in\mathcal{V}}p(v\mid y_{[i-1]})^\gamma} = p(v\mid y_{[i-1]}).$$

But this would imply for every $v_s, v_r \in \mathcal{V}$ and for every $y_{[i-1]} \in \mathcal{Y}_{[i-1]}$ we have

$$\frac{p(v_s\mid y_{[i-1]})}{p(v_r\mid y_{[i-1]})} = \frac{\frac{p(v_s|y_{[i-1]})^\gamma}{\sum_{v_j\in\mathcal{V}}p(v_j|y_{[i-1]})^\gamma}}{\frac{p(v_r|y_{[i-1]})^\gamma}{\sum_{v_j\in\mathcal{V}}p(v_j|y_{[i-1]})^\gamma}} = \left(\frac{p(v_s\mid y_{[i-1]})}{p(v_r\mid y_{[i-1]})}\right)^\gamma.$$

Thus, if $\gamma \neq 1$, then we must have $\frac{p(v_s|y_{[i-1]})}{p(v_r|y_{[i-1]})} \in \{0, 1, \infty\}$ for this to be true. Seeing this needs to happen for every for every $v_s, v_r,$, this would imply the next-token distribution is a uniform distribution or a deterministic distribution. □

B.2.15 Proof of Proposition 8

Proof. By log properties and linearity of expectation:

$$\mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\prod_{i=1}^{L}\frac{p(y_i \mid y_{[i-1]})^\gamma}{\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma}\right)\right] = \mathop{\mathbb{E}}_{y\sim p}\left[-\sum_{i=1}^{L}\log\left(\frac{p(y_i \mid y_{[i-1]})^\gamma}{\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma}\right)\right] = \sum_{i=1}^{L}\mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\frac{p(y_i \mid y_{[i-1]})^\gamma}{\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma}\right)\right] \tag{1}$$

We will only look at one of these expectations in the sum. Choose $j \in [L]$. Then:

$$\mathop{\mathbb{E}}_{y\sim p}\left[-\log\left(\frac{p(y_i \mid y_{[i-1]})^\gamma}{\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma}\right)\right] = \mathop{\mathbb{E}}_{y\sim p}\left[-\gamma\log\Big(p(y_i \mid y_{[i-1]})\Big) + \log\left(\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma\right)\right] \tag{$\star$}$$

Now, we have the following inequalities:

$$\begin{aligned}
\log\left(\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma\right) &\le \log\left(|V|\max_{y_j\in\mathcal{V}}\Big\{p(y_j \mid y_{[i-1]})^\gamma\Big\}\right) \\
&= \log(|V|) + \gamma\max_{y_j\in\mathcal{V}}\{\log\Big(p(y_j \mid y_{[i-1]})\Big)\} \\
\log\left(\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma\right) &\ge \log\left(|V|\min_{y_j\in\mathcal{V}}\Big\{p(y_j \mid y_{[i-1]})^\gamma\Big\}\right) \\
&= \log(|V|) + \gamma\min_{y_j\in\mathcal{V}}\{\log\Big(p(y_j \mid y_{[i-1]})\Big)\} \\
\log\left(\sum_{y_j\in\mathcal{V}} p(y_j \mid y_{[i-1]})^\gamma\right) &\ge \log\left(\max_{y_j\in\mathcal{V}}\Big\{p(y_j \mid y_{[i-1]})^\gamma\Big\}\right) \\
&= \gamma\max_{y_j\in\mathcal{V}}\{\log\Big(p(y_j \mid y_{[i-1]})\Big)\}
\end{aligned}$$

Using these, notice:

$$(\star) \le \mathop{\mathbb{E}}_{y\sim p}\left[-\gamma\log\Big(p(y_i \mid y_{[i-1]})\Big) + \log(|V|) + \gamma\max_{y_j\in\mathcal{V}}\{\log\Big(p(y_j \mid y_{[i-1]})\Big)\}\right] = \gamma C_1 + \log(|V|) \tag{2}$$

where $C_{1,i} \in \mathbb{Z}_+$ is a constant that only depends on p.

For the lower bound, by just substituting the other inequalities in we get:

$$(*) \geq -\gamma C_{2,i} + \log(|V|) \tag{3}$$

$$(*) \geq \gamma C_{3,i} \tag{4}$$

where $C_{2,i}, C_{3,i} \in \mathbb{Z}_+$ are constants that only depend on p.

Substituting these inequalities back into (1) will give us what we wanted to show. □

We assumed $\mathcal{Y} = \mathcal{V}^L$ to allow us to use the middle inequality of the three. If we do not asumme this, then it is possible $\min_{y_j \in \mathcal{V}}\{p(y_j \mid y_{[i-1]})^\gamma\} = 0$. To then use this inequality, we would need $|V|$ to be replaced with the amount of tokens with a non-zero probability, $|V_{y_{[i-1]}}|$. This would then require taking the expectation over $\log(|V_{y_{[i-1]}}|)$ to get our bounds. We could also get a matching upper bound by doing the same with the upper bound.

APPENDIX C

Appendix for Chapter 4

C.1 Proving the Relationship Between $VC(\ell \circ \mathcal{H})$ and $GNdim(\mathcal{H}, \ell)$

Here we show that the VC-dimenion of the loss class and the Generalized Natarajan Dimension are bounded by each other.

Lemma C.1.1. *Given a $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem, we have*

$$GNdim(\mathcal{H}, \ell) \leq VC(\ell \circ \mathcal{H}) \leq 4.67 GNdim(\mathcal{H}, \ell) \log_2(|\sigma(Z)| + 1).$$

Proof. First we will prove $GNdim(\mathcal{H}, \ell) \leq VC(\ell \circ \mathcal{H})$. Let S, $|S| = n$ GN-shatter $\mathcal{H}, \ell$. Then $\exists f_1, f_2 \in \mathcal{H}$ such that $\forall x \in S$, $\sigma(f_1(x)) \neq \sigma(f_2(x))$ and $\forall T \subset S$, $\exists h_T \in \mathcal{H}$ where $\forall x \in T$, $\sigma(h_T(x)) = \sigma(f_1(x))$ and $\forall x \in S \setminus T$, $\sigma(h_T(x)) = \sigma(f_2(x))$. Let $S' = \{(x, y) \mid x \in S, y \in \sigma(f_1(x)) \triangle \sigma(f_2(x))\}$ where $\triangle$ denotes the symmetric difference of the two sets. Using f_1, f_2 and h_T as above we can see that $\ell \circ \mathcal{H}$ is VC-shattered by S'.

Now we will prove $VC(\ell \circ \mathcal{H}) \leq 4.67 GNdim(\mathcal{H}, \ell) \log_2(|\sigma(Z)| + 1))$. It is easy to see that for any set $S \subset \mathcal{X} \times \mathcal{Y}$ we have $|(\ell \circ \mathcal{H})(S)| \leq |\sigma(\mathcal{H})(S_{\mathcal{X}})|$. This is because if $\ell \circ h$ and $\ell \circ h'$ differ on a point, then h, h' must be different on that point as well.

Note how $\sigma(\mathcal{H})$ only has $|\sigma(\mathcal{Z})|$ different possible outputs. By Ben-David et al. [1995], we have for m samples:

$$|\sigma \circ \mathcal{H}| \leq \left(\frac{me(|\sigma(\mathcal{Z})| + 1)^2}{2Ndim(\sigma \circ \mathcal{H})} \right)^{Ndim(\sigma \circ \mathcal{H})}$$

Which, by 4.4.0.2, we have

$$\left(\frac{me(|\sigma(\mathcal{Z})| + 1)^2}{2Ndim(\sigma \circ \mathcal{H})} \right)^{Ndim(\sigma \circ \mathcal{H})} = \left(\frac{me(|\sigma(\mathcal{Z})| + 1)^2}{2GNdim(\mathcal{H}, \ell)} \right)^{GNdim(\mathcal{H}, \ell)}$$

Thus, for a sample of size $VC(\ell \circ \mathcal{H})$ to shatter $\ell \circ \mathcal{H}$, we have:

$$2^{VC(\ell \circ \mathcal{H})} \leq |\ell \circ \mathcal{H}| \leq |\sigma \circ \mathcal{H}| \leq \left(\frac{VC(\ell \circ \mathcal{H}) e (|\sigma(\mathcal{Z})| + 1)^2}{2GNdim(\mathcal{H}, \ell)} \right)^{GNdim(\mathcal{H}, \ell)}$$

This is the same inequality set-up as seen in Ben-David et al. [1995], thus we get the same bounds, except we substitute in $GNdim(\mathcal{H}, \ell)$ and $|\sigma(\mathcal{Z})|$ to get

$$VC(\ell \circ \mathcal{H}) \leq 4.67 GNdim(\mathcal{H}, \ell) \log_2(|\sigma(Z)| + 1)$$

□

Given Thoerem 4.4.1 and our assumption of $|\sigma(\mathcal{Z})| < \infty$ we have the following corollary:

Corollary C.1.1.1. *Given a $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem studied in this chapter, we have*

$$(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell) \text{ is learnable } \iff VC(\ell \circ \mathcal{H}) < \infty$$

C.2 Finite $GNdim(\mathcal{H}, \ell)$ is Necessary for Learnability Proof

C.2.1 Proof by Adapting The No Free Lunch Theorem

Proof. We will show the contrapositive. Let us have a $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning problem with $GNdim(\mathcal{H}, \ell) = \infty$. Then, by shattering, for any m we can find a set of $2m$ points where $f_1, f_2, h \in \mathcal{H}$ such that $\sigma(f_1(x)) \neq \sigma(f_2(x))$ for all $x \in \{x_1, \ldots, x_{2m}\}$ but $\sigma(h(x)) = \sigma(f_1(x))$ for $x \in \{x_1, \ldots, x_m\}$ and $\sigma(h(x)) = \sigma(f_2(x))$ for the others. Let $T = 2^{2m}$, and $h_1, ..., h_T$ be all of the functions that are guaranteed to exists by the shattering condition.

Let us define the distribution D_{f_1} as follows:

$$\mathbb{P}_{D_{f_1}}(x) = \begin{cases} 1/2m & x \in \{x_1, \ldots, x_{2m}\} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbb{P}_{D_{f_1}}(y \mid x) = \begin{cases} 1/|\sigma(f_1(x))| & y \in \sigma(f_1(x)) \\ 0 & \text{otherwise} \end{cases}$$

Notice that $\mathbb{P}_{D_{f_1}}$ is a uniform distribution over our $2m$ inputs ($D_{f1}^{\mathcal{X}}$), and the output values are also uniformly distributed over the possible values where f_1 would get 0 loss on ($D_{f1}^{\mathcal{Y}|\mathcal{X}}$). Let us define D_i, $D_i^{\mathcal{Y}|\mathcal{X}}$, and $D_i^{\mathcal{X}}$ the same as above, but for h_i instead of f_1. Note each of these D_i distributions are realizable under $\mathcal{H}$.

Let $S_{X_1}, ..., S_{X_k}$ denote all of the samples of size m sampled from $D_i^{\mathcal{X}}$. Note k is the same for every i as this is just all samples of size m from $\mathcal{X}$. We will denote S^i_{Y,X_j} as the full (x, y) samples and S^i_Y will be the $\mathcal{Y}$ values from sample S^i_{Y,X_j}. While there is an implicit X_j needed for S^i_Y, we drop it to ease notation as we do not ever look at two S^i_Y with different X_js at the same time. Since each sample of S_{X_j} of size m is equally likely to be chosen, we can use the law of total expectation to get

$$\mathbb{E}_{S\sim D_i}[L_{D_i}(A(S))] = \frac{1}{k}\sum_{j=1}^{k} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))]$$

where the expectation on the right hand side is over all possible label values we could get from our input sample X_j. Thus we get

$$\begin{aligned}
&\max_{i\in[T]} \mathbb{E}_{S\sim D_i}[L_{D_i}(A(S))] = \max_{i\in[T]} \frac{1}{k}\sum_{j=1}^{k} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))] \geq \\
&\frac{1}{T}\sum_{i=1}^{T}\frac{1}{k}\sum_{j=1}^{k} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))] = \\
&\frac{1}{k}\sum_{j=1}^{k}\frac{1}{T}\sum_{i=1}^{T} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))] \geq \\
&\min_{j\in[k]} \frac{1}{T}\sum_{i=1}^{T} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))]
\end{aligned}$$

Let us fix a $j \in [k]$, and let $\{s_1, ...s_p\} = \mathcal{X} \setminus S^i_{X_j}$. Notice $p \geq m$ as $|\mathcal{X}| = 2m$ and $|S^i_{X_j}| \leq m$. We then get for any function $f : \mathcal{X} \to \mathcal{Z}$

$$\begin{aligned}
&L_{D_i}(f) = \frac{1}{2m}\sum_{r=1}^{2m} \mathop{\mathbb{E}}_{v\sim Unif(\sigma(h_i(x_r)))}\left[1_{f(x_r)\notin\tau(v)}\right] \geq \\
&\frac{1}{2p}\sum_{r=1}^{p} \mathop{\mathbb{E}}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{f(s_r)\notin\tau(v)}\right]
\end{aligned}$$

Thus, as expectation is a linear operator:

$$\begin{aligned}
&\frac{1}{T}\sum_{i=1}^{T} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}[L_{D_i}(A(S^i_{Y,X_j}))] \\
&\geq \frac{1}{T}\sum_{i=1}^{T}\frac{1}{2p}\sum_{r=1}^{p} \mathbb{E}_{S^i_Y \sim D_i^{\mathcal{Y}|S_{X_j}}}\left[\mathop{\mathbb{E}}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]\right].
\end{aligned}$$

Continuing, we see

$$\frac{1}{T}\sum_{i=1}^{T}\frac{1}{2p}\sum_{r=1}^{p}\mathbb{E}_{S_Y^i\sim D_i^{\mathcal{Y}|S_{X_j}}}\left[\mathbb{E}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]\right] =$$
$$\frac{1}{2p}\sum_{r=1}^{p}\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}_{S_Y^i\sim D_i^{\mathcal{Y}|S_{X_j}}}\left[\mathbb{E}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]\right] \geq$$
$$\frac{1}{2}\min_{r\in[p]}\frac{1}{T}\sum_{i=1}^{T}\mathbb{E}_{S_Y^i\sim D_i^{\mathcal{Y}|S_{X_j}}}\left[\mathbb{E}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]\right].$$

Fix $r\in[p]$. Partition $h_1,...,h_T$ into disjoint tuples $(h_i,h_{i'})$ by $\sigma(h_i(x))\neq\sigma(h_{i'}(x))$ iff $x=s_r$. Since s_r is not in S_{X_j}, we can see that each pair has the same probabilities of getting the same S_Y^i given S_{X_j}. When $S^i_{Y,X_j}=S^{i'}_{Y,X_j}$, for each tuple we have that for exactly at most one of them (without loss of generality, assume its h_i)

$$\mathbb{E}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]=0.$$

We have this for at most one i as we know that there are no strict subsets in $\sigma(\mathcal{Z})$. Thus, WLOG, if $\sigma(A(S^i_{Y,X_j})(s_r))=\sigma(h_i(s_r))$, then $\exists v\in\sigma(h_{i'}(s_r))$ where $v\notin\sigma(A(S^i_{Y,X_j})(s_r))$.

Therefore:

$$\mathbb{E}_{v\sim Unif(\sigma(h_{i'}(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]=\frac{|\sigma(h_{i'}(s_r)))\setminus\sigma(A(S^i_j)(s_r))|}{|\sigma(h_{i'}(s_r)))|}\geq\frac{1}{|\sigma(h_{i'}(s_r)))|}$$
$$\geq\frac{1}{\max_{z\in\mathcal{Z}^C}|\sigma(z)|}$$

With this, we see that over every tuple

$$\mathbb{E}_{v\sim Unif(\sigma(h_i(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]+\mathbb{E}_{v\sim Unif(\sigma(h_{i'}(s_r)))}\left[1_{A(S^i_{Y,X_j})(s_r)\notin\tau(v)}\right]\geq\frac{1}{\max_{z\in\mathcal{Z}^C}|\sigma(z)|}.$$

We note again that S^i_{Y,X_j} has the same probability as $S^{i'}_{Y,X_j}$ in each respective tuple. Thus,

we can substitute the expectations over $D_i^{\mathcal{Y}|S_{X_j}}$ and $D_{i'}^{\mathcal{Y}|S_{X_j}}$ into just one $D_i^{\mathcal{Y}|S_{X_j}}$ to yield

$$\mathbb{E}_{S_Y^i \sim D_i^{\mathcal{Y}|S_{X_j}}} \left[\mathbb{E}_{v \sim Unif(\sigma(h_i(s_r)))} \left[1_{A(S^i_{Y,X_j})(s_r) \notin \tau(v)} \right] \right] +$$
$$\mathbb{E}_{S_Y^{i'} \sim D_{i'}^{\mathcal{Y}|S_{X_j}}} \left[\mathbb{E}_{v \sim Unif(\sigma(h_{i'}(s_r)))} \left[1_{A(S^{i'}_{Y,X_j})(s_r) \notin \tau(v)} \right] \right] =$$
$$\mathbb{E}_{S^i_{j,Y} \sim D_i^{\mathcal{Y}|S_{X_j}}} \left[\mathbb{E}_{v \sim Unif(\sigma(h_i(s_r)))} \left[1_{A(S^i_{Y,X_j})(s_r) \notin \tau(v)} \right] + \mathbb{E}_{v \sim Unif(\sigma(h_{i'}(s_r)))} \left[1_{A(S^i_{Y,X_j})(s_r) \notin \tau(v)} \right] \right]$$
$$\geq \frac{1}{\max_{z \in \mathcal{Z}^C} |\sigma(z)|}.$$

By summing each partition together we see

$$\frac{1}{T} \sum_{i=1}^{T} \mathbb{E}_{S^i_{j,Y} \sim D_i^{\mathcal{Y}|S^i_{j,X}}} \left[\mathbb{E}_{v \sim Unif(\sigma(h_i(s_r)))} \left[1_{A(S^i_{Y,X_j})(s_r) \notin \tau(v)} \right] \right]$$
$$\geq \frac{1}{T} \left(\frac{T}{2} \frac{1}{\max_{z \in \mathcal{Z}^C} |\sigma(z)|} \right) = \frac{1}{2 \max_{z \in \mathcal{Z}^C} |\sigma(z)|}.$$

Therefore,

$$\max_{i \in [T]} \mathbb{E}_{S \sim D_i}[L_{D_i}(A(S))] \geq \frac{1}{2} \min_{r \in [p]} \frac{1}{T} \sum_{i=1}^{T} \mathbb{E}_{S^i_{j,Y} \sim D_i^{\mathcal{Y}|S_{X_j}}} \left[\mathbb{E}_{v \sim Unif(\sigma(h_i(s_r)))} \left[1_{A(S^i_{Y,X_j})(s_r) \notin \tau(v)} \right] \right] \geq$$
$$\frac{1}{2} \min_{r \in [p]} \frac{1}{2 \max_{z \in \mathcal{Z}^C} |\sigma(z)|} = \frac{1}{4 \max_{z \in \mathcal{Z}^C} |\sigma(z)|}.$$

Thus, we have shown there exists a realizable distribution where $\mathcal{A}$ does not do well, thereby showing this is not PAC-learnable, which is what we needed to show for the contrapositive. □

C.3 Finite $GNdim(\mathcal{H}, \ell)$ is Sufficient for Learnability Proof

C.3.1 Using VC Dimension of Loss Class

Proof. Suppose we have an $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ learning scenario where $GNdim(\mathcal{H}, \ell)$ is finite. By Lemma C.1.1, we have that

$$VC(\ell \circ \mathcal{H}) \leq 4.67 GNdim(\mathcal{H}, \ell) \log_2(|\sigma(Z)| + 1).$$

Since $|\sigma(\mathcal{Z})| < \infty$, we know that $VC(\ell \circ \mathcal{H})$ must be finite. It is known that the upper bound of the sample complexity of $\ell \circ \mathcal{H}$ is

$$O\left(\frac{VC(\ell \circ \mathcal{H}) + \log(1/\delta)}{\epsilon^2}\right)$$

due to it being a binary classifier. Notice for any distribution on $\mathcal{X} \times \mathcal{Y}$, there is an equivalent distribution over $\mathcal{X} \times \mathcal{Y} \times \{0\}$. Therefore, learnability of the 0-1 loss of $\ell \circ \mathcal{H}$ on the latter distribution implies learnability of $\mathcal{H}$ with ℓ on the former. This gives us an upper bound on the sample complexity needed to learn $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ by replacing $VC(\ell \circ \mathcal{H})$ with $4.67 GNdim(\mathcal{H}, \ell) \log_2(|\sigma(Z)| + 1)$. Thus, we have learnability of $(\mathcal{X}, \mathcal{Z}, \mathcal{Y}, \mathcal{H}, \ell)$ if $GNdim(\mathcal{H}, \ell)$ is finite.

□

C.3.2 Special Case where $\mathcal{Z} = \mathcal{Y}$

Proof. Much like Lemma 4.4.2, we shall use the $(\mathcal{X}, \mathcal{Y}^C, \mathcal{H}^C, \ell^C)$ equivalent learning problem.

Suppose $(\mathcal{H}, \ell)$ has finite generalized Natarajan dimension. For each $y \in \mathcal{Y}^C$, put an arbitrary ordering on $\sigma(y)$, and then for all $k > |\sigma(y)|$, let $\sigma(y)_k$ be some label not in $\mathcal{Y}^C$ (from here on out denoted 0). Let the maps $\sigma_i : \mathcal{Y}^C \cup \{0\} \to \mathcal{Y}^C \cup \{0\}$ denote the function $\sigma_i(y) = \sigma(y)_i$, where $\sigma_i(0) = 0 \ \forall i$. We can extend this map (abusing notation) to $\sigma_i : \mathcal{X} \times (\mathcal{Y}^C \cup \{0\}) \to \mathcal{X} \times (\mathcal{Y}^C \cup \{0\})$, $\sigma_i(x, y) = (x, \sigma_i(y))$. Now, given a distribution D, let us define a new distribution $D^i := D \circ \sigma_i^{-1}$ by the pushforward of D through σ_i. Thus D^i can be thought of as the distribution where each y "becomes" $\sigma(y)_i$. Notice then

$$1 - L_{\mathcal{D}}(h) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[1_{h(x)\in\sigma(y)}\right] = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\sum_{i=1}^{|\sigma(y)|} 1_{h(x)=\sigma(y)_i}\right] =$$
$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\sum_{i=1}^{|\sigma(y)|} 1_{h(x)=\sigma(y)_i} + \sum_{i=|\sigma(y)|+1}^{k} 0\right] = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\sum_{i=1}^{|\sigma(y)|} 1_{h(x)=\sigma(y)_i} + \sum_{i=|\sigma(y)|+1}^{k} 1_{h(x)=0}\right] =$$
$$\sum_{i=1}^{k} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[1_{h(x)=\sigma(y)_i}\right] = \sum_{i=1}^{k} \mathop{\mathbb{E}}_{(x,y')\sim\mathcal{D}^i}\left[1_{h(x)=y'}\right] = \sum_{i=1}^{k} 1 - L_{\mathcal{D}^i}(h)$$

Where

$$L_{\mathcal{D}^i}(h) := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}^i}\left[1_{h(x)\neq y}\right]$$

is the 0-1 loss for a multiclass learning problem on the distribution of D^i, and where the penultimate equality comes from

$$
\begin{aligned}
&D(\{(x,y) \mid h(x) = \sigma_i(y)\}) = \\
&D(\{(x,y) \mid h(x) = y' \wedge y' = \sigma_i(y)\}) = \\
&D(\{(x,y) \mid h(x) = y' \wedge \sigma_i(x,y) = (x,y')\}) = \\
&D \circ \sigma_i^{-1}(\{(x,y') \mid h(x) = y'\}) = \\
&D^i(\{(x,y') \mid h(x) = y'\})
\end{aligned}
$$

For the sample loss of a set of size m, we can do a similar decomposition.

$$
\begin{aligned}
&1 - L_S(h) = \frac{|\{(y_j, h(x_j)) \in C\}|}{m} = \frac{\sum_{j=1}^{m} 1_{h(x_j) \in \sigma(y_j)}}{m} = \\
&\frac{\sum_{j=1}^{m} \sum_{i=1}^{k} 1_{h(x_j) = \sigma(y_j)_i}}{m} = \frac{\sum_{i=1}^{k} \sum_{j=1}^{m} 1_{h(x_j) = \sigma(y_j)_i}}{m} = \sum_{i=1}^{k} 1 - L_{S^i}(h)
\end{aligned}
$$

Using these decompositions, we get

$$
\begin{aligned}
&|L_D(h) - L_S(h)| = |L_D(h) - 1 + 1 - L_S(h)| = \\
&\left| -\sum_{i=1}^{k} 1 - L_{D^i}(h) + \sum_{i=1}^{k} 1 - L_{S^i}(h) \right| = \left| \sum_{i=1}^{k} L_{D^i}(h) - L_{S^i}(h) \right| \leq \sum_{i=1}^{k} |L_{D^i}(h) - L_{S^i}(h)|
\end{aligned}
$$

Since we have finite generalized Natarajan dimension, by corollary 4.4.0.2 we have that $\mathcal{H}^C$ is multiclass learnable, and thus has the uniform convergence property with function $m^{UC}_{\mathcal{H}^C}$ (Shalev-Shwartz and Ben-David [2014b]). Since each L_{D^i}, L_{S^i} are equivalent to a multiclass problem on a different distribution, pick sample S with $|S| \geq m^{UC}_{\mathcal{H}^C}(\epsilon/k, \delta/k)$. By union bound we get:

$$
\begin{aligned}
&\mathbb{P}\left(\sup_{h \in \mathcal{H}} |L_D(h) - L_S(h)| > \epsilon \right) \leq \mathbb{P}\left(\sup_{h \in \mathcal{H}} \sum_{i=1}^{k} \left| L^i_D(h) - L^i_S(h) \right| > \epsilon \right) \leq \\
&\mathbb{P}\left(\sum_{i=1}^{k} \sup_{h \in \mathcal{H}} \left| L^i_D(h) - L^i_S(h) \right| > \epsilon \right) \leq \mathbb{P}\left(\bigcup_{i=1}^{k} \left(\sup_{h \in \mathcal{H}} \left| L^i_D(h) - L^i_S(h) \right| > \epsilon/k \right) \right) \leq k\delta/k = \delta
\end{aligned}
$$

Therefore, we have that $\mathcal{H}^C$ has the uniform convergence property and thus ERM is a valid learner for the problem. □

C.4 Miscellaneous

C.4.1 Example of a Learning Problem where $GNdim(\mathcal{H}, \ell) = \infty$ but it is Learnable

Let $\mathcal{X}$ be an infinite input space, $\mathcal{Y} = \{1, 2, 3\}$, $\mathcal{H} = \{1, 2\}^{\mathcal{X}} \cup \{3\}^{\mathcal{X}}$, and let the loss matrix be as below:

$$L = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

where $\ell(i, j) = L_{ij}$. Note how $GNdim(\mathcal{H}, \ell) = VC(\ell \circ \mathcal{H}) = \infty$ as we have all functions from $\mathcal{X}$ to $\{1, 2\}$, but the algorithm that always chooses $h \in \mathcal{H}$ such that $\forall x \in \mathcal{X}$, $h(x) = 3$ will always be a valid PAC-learner.

BIBLIOGRAPHY

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.

Noga Alon, Steve Hanneke, Ron Holzman, and Shay Moran. A theory of pac learnability of partial concept classes. In 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS), pages 658–671. IEEE, 2022.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016.

Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. arXiv preprint arXiv:2403.06963, 2024.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. Journal of the American Statistical Association, 101(473):138–156, March 2006. ISSN 0162-1459. doi: 10.1198/016214505000000907.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. Advances in neural information processing systems, 30, 2017.

S Ben-David, N Cesabianchi, D Haussler, and PM Long. Characterizations of learnability for classes of {O,..., n}-valued functions. Journal of Computer and System Sciences, 50 (1):74–86, 1995.

Marco Bressan, Nataly Brukhim, Nicolò Cesa-Bianchi, Emmanuel Esposito, Yishay Mansour, Shay Moran, and Maximilian Thiessen. A fine-grained characterization of pac learnability. In Nika Haghtalab and Ankur Moitra, editors, Proceedings of Thirty Eighth Conference on Learning Theory, volume 291 of Proceedings of Machine Learning Research, pages 641–676. PMLR, 30 Jun–04 Jul 2025. URL https://proceedings.mlr.press/v291/bressan25b.html.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. Advances in Neural Information Processing Systems, 33: 1877–1901, 2020.

Nataly Brukhim, Daniel Carmon, Irit Dinur, Shay Moran, and Amir Yehudayoff. A characterization of multiclass learnability. In 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS), pages 943–955. IEEE, 2022.

Moses Charikar and Chirag Pabbaraju. A characterization of list learnability. In Proceedings of the 55th Annual ACM Symposium on Theory of Computing, pages 1713–1726, 2023.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. Journal of Machine Learning Research, 24(240):1–113, 2023.

Amit Daniely and Shai Shalev-Shwartz. Optimal learners for multiclass problems. In Conference on Learning Theory, pages 287–316. PMLR, 2014.

Amit Daniely, Sivan Sabato, Shai Ben-David, and Shai Shalev-Shwartz. Multiclass learnability and the erm principle. J. Mach. Learn. Res., 16(1):2377–2404, 2015.

Ofir David, Shay Moran, and Amir Yehudayoff. On statistical learning via the lens of compression. arXiv preprint arXiv:1610.03592, 2016.

Krzysztof Dembczyński, Willem Waegeman, Weiwei Cheng, and Eyke Hüllermeier. Regret analysis for performance metrics in multi-label classification: The case of hamming and subset zero-one loss. In José Luis Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, Machine Learning and Knowledge Discovery in Databases, page 280–295, Berlin, Heidelberg, 2010. Springer. ISBN 978-3-642-15880-3. doi: 10.1007/978-3-642-15880-3_24.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), pages 4171–4186, 2019.

Shehzaad Dhuliawala, Ilia Kulikov, Ping Yu, Asli Celikyilmaz, Jason Weston, Sainbayar Sukhbaatar, and Jack Lanchantin. Adaptive decoding via latent preference optimization. arXiv preprint arXiv:2411.09661, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

Richard M Dudley. The sizes of compact subsets of hilbert space and continuity of gaussian processes. Journal of Functional Analysis, 1(3):290–330, 1967.

Benjamin L Edelman, Surbhi Goel, Sham Kakade, and Cyril Zhang. Inductive biases and variable creation in self-attention mechanisms. In International Conference on Machine Learning, pages 5793–5831. PMLR, 2022.

Dylan J Foster and Alexander Rakhlin. l∞ vector contraction for rademacher complexity. arXiv preprint arXiv:1911.06468, 6, 2019.

Hengyu Fu, Tianyu Guo, Yu Bai, and Song Mei. What can a single attention layer learn? a study through the random features lens. arXiv preprint arXiv:2307.11353, 2023.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of llm inference using lookahead decoding. arXiv preprint arXiv:2402.02057, 2024.

Wei Gao and Zhi-Hua Zhou. On the consistency of multi-label learning. In Proceedings of the 24th Annual Conference on Learning Theory, page 341–358. JMLR Workshop and Conference Proceedings, December 2011. URL https://proceedings.mlr.press/v19/gao11a.html.

Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In International Conference on Machine Learning, pages 3419–3430. PMLR, 2020.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In Conference On Learning Theory, pages 297–299. PMLR, 2018.

Yann Guermeur. Vc theory of large margin multi-category classifiers. The Journal of Machine Learning Research, 8:2551–2594, 2007.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948, 2025.

Steve Hanneke, Shay Moran, and Waknine Tom. List sample compression and uniform convergence. In The Thirty Seventh Annual Conference on Learning Theory, pages 2360–2388. PMLR, 2024.

Steve Hanneke, Qinglin Meng, and Amirreza Shaeiri. Representation preserving multiclass agnostic to realizable reduction. In Forty-second International Conference on Machine Learning, 2025.

Max Hopkins, Daniel M Kane, Shachar Lovett, and Gaurav Mahajan. Realizable learning is all you need. In Conference on Learning Theory, pages 3015–3069. PMLR, 2022.

Petr Hurtik, Stefania Tomasiello, Jan Hula, and David Hynar. Binary cross-entropy with dynamical clipping. Neural Computing and Applications, 34(14):12029–12041, 2022.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. arXiv preprint arXiv:2412.16720, 2024.

Sham M Kakade, Karthik Sridharan, and Ambuj Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. Advances in neural information processing systems, 21, 2008.

Alkis Kalavasis, Grigoris Velegkas, and Amin Karbasi. Multiclass learnability beyond the pac framework: Universal rates and partial concept classes. Advances in Neural Information Processing Systems, 35:20809–20822, 2022.

Daniel Martin Katz, Michael James Bommarito, Shang Gao, and Pablo Arredondo. Gpt-4 passes the bar exam. Available at SSRN 4389233, 2023.

Aryeh Kontorovich and Idan Attias. Fat-shattering dimension of k-fold maxima. arXiv preprint arXiv:2110.04763, 2021.

Oluwasanmi O Koyejo, Nagarajan Natarajan, Pradeep K Ravikumar, and Inderjit S Dhillon. Consistent multilabel classification. Advances in Neural Information Processing Systems, 28, 2015.

Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: A llvm-based python jit compiler. In Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, pages 1–6, 2015.

Michel Ledoux and Michel Talagrand. Probability in Banach Spaces: isoperimetry and processes, volume 23. Springer Science & Business Media, 1991.

Yingcong Li, Yixiao Huang, Muhammed E Ildiz, Ankit Singh Rawat, and Samet Oymak. Mechanics of next token prediction with self-attention. In International Conference on Artificial Intelligence and Statistics, pages 685–693. PMLR, 2024.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In Text summarization branches out, pages 74–81, 2004.

Pengxiao Lin, Zhongwang Zhang, and Zhi-Qin John Xu. Reasoning bias of next token prediction training. arXiv preprint arXiv:2502.02007, 2025.

Shan Lin and Jingwei Zhang. Generalization bounds for convolutional neural networks. arXiv preprint arXiv:1910.01487, 2019.

Liping Liu and Thomas Dietterich. Learnability of the superset label learning problem. In International conference on machine learning, pages 1629–1637. PMLR, 2014.

Balas K Natarajan. On learning sets and functions. Machine Learning, 4(1):67–97, 1989.

Gerhard Paaß and Sven Giesselbach. Foundation models for natural language processing: Pre-trained language models integrating media. Springer Nature, 2023.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting of the Association for Computational Linguistics, pages 311–318, 2002.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? arXiv preprint arXiv:1909.01066, 2019.

Julius Pettersson and Petter Falkman. Comparison of lstm, transformers, and mlp-mixer neural networks for gaze based human intention prediction. Frontiers in Neurorobotics, 17:1157957, 2023.

Gilles Pisier. Remarques sur un résultat non publié de b. maurey. Séminaire d'Analyse fonctionnelle (dit" Maurey-Schwartz"), pages 1–12, 1981.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Vinod Raman, Unique Subedi, and Ambuj Tewari. On the learnability of multilabel ranking. Advances in Neural Information Processing Systems, 36:11988–11999, 2023.

Vinod Raman, Unique Subedi, and Ambuj Tewari. A characterization of multioutput learnability. Journal of Machine Learning Research, 25(342):1–54, 2024a.

Vinod Raman, Unique Subedi, and Ambuj Tewari. Online learning with set-valued feedback. In The Thirty Seventh Annual Conference on Learning Theory, pages 4381–4412. PMLR, 2024b.

Harish G Ramaswamy and Shivani Agarwal. Convex calibration dimension for multiclass loss matrices. Journal of Machine Learning Research, 17(14):1–45, 2016.

Harish G Ramaswamy, Shivani Agarwal, and Ambuj Tewari. Convex calibrated surrogates for low-rank loss matrices with applications to subset ranking losses. Advances in Neural Information Processing Systems, 26, 2013.

Nikunj Saunshi, Sadhika Malladi, and Sanjeev Arora. A mathematical exploration of why language models help solve downstream tasks. In International Conference on Learning Representations, 2021. URL https://openreview.net/forum?id=vVjIW3sEc1s.

S. Shalev-Shwartz and S. Ben-David. Understanding Machine Learning: From Theory to Algorithms. Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, 2014a. ISBN 9781107057135. URL https://books.google.com/books?id=ttJkAwAAQBAJ.

Shai Shalev-Shwartz and Shai Ben-David. Understanding machine learning: From theory to algorithms. Cambridge University Press, 2014b.

Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. A thorough examination of decoding methods in the era of llms. arXiv preprint arXiv:2402.06925, 2024.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. arXiv preprint arXiv:2408.03314, 2024.

Ambuj Tewari and Peter L. Bartlett. On the consistency of multiclass classification methods. Journal of Machine Learning Research, 8(36):1007–1025, 2007. URL http://jmlr.org/papers/v8/tewari07a.html.

Christos Thrampoulidis. Implicit optimization bias of next-token prediction in linear models. arXiv preprint arXiv:2402.18551, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971, 2023.

Lan V Truong. On rademacher complexity-based generalization bounds for deep learning. arXiv preprint arXiv:2208.04284, 2022.

Leslie G Valiant. A theory of the learnable. Communications of the ACM, 27(11):1134–1142, 1984.

Vladimir Vapnik and Alexey Chervonenkis. Theory of pattern recognition, 1974.

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.

Vladimir Vovk. The Fundamental Nature of the Log Loss Function, volume 9300 of Lecture Notes in Computer Science, page 307–318. Springer International Publishing, Cham, 2015. ISBN 978-3-319-23533-2. doi: 10.1007/978-3-319-23534-9_20. URL https://link.springer.com/10.1007/978-3-319-23534-9_20.

Qi Wang, Yue Ma, Kun Zhao, and Yingjie Tian. A comprehensive survey of loss functions in machine learning. Annals of Data Science, 9(2):187–212, 2022.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. arXiv preprint arXiv:1906.01787, 2019.

Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: a case study on approximating turing machines with transformers. Advances in Neural Information Processing Systems, 35:12071–12083, 2022.

Hongxin Wei, Huiping Zhuang, Renchunzi Xie, Lei Feng, Gang Niu, Bo An, and Yixuan Li. Mitigating memorization of noisy labels by clipping the model prediction. 2023.

Gian Wiher, Clara Meister, and Ryan Cotterell. On decoding strategies for neural text generators. Transactions of the Association for Computational Linguistics, 10:997–1012, 2022.

Guoqiang Wu and Jun Zhu. Multi-label classification: do hamming loss and subset accuracy really conflict with each other? In Advances in Neural Information Processing Systems, volume 33, page 3130–3140. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/20479c788fb27378c2c99eadcf207e7f-Abstract.html.

Huijuan Wu, Keqilao Meng, Daoerji Fan, Zhanqiang Zhang, and Qing Liu. Multistep short-term wind speed forecasting using transformer. Energy, 261:125231, 2022.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. Advances in Neural Information Processing Systems, 32, 2019.

Nianzu Yang, Huaijin Wu, Kaipeng Zeng, Yang Li, Siyuan Bao, and Junchi Yan. Molecule generation for drug design: a graph learning perspective. Fundamental Research, 2024.

Tong Zhang. Covering number bounds of certain regularized linear function classes. Journal of Machine Learning Research, 2(Mar):527–550, 2002.

Yufeng Zhang, Boyi Liu, Qi Cai, Lingxiao Wang, and Zhaoran Wang. An analysis of attention via the lens of exchangeability and latent variable models. arXiv preprint arXiv:2212.14852, 2022.

Jianing Zhou and Suma Bhat. Paraphrase generation: A survey of the state of the art. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 5075–5086, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.414. URL https://aclanthology.org/2021.emnlp-main.414/.

Wenhong Zhu, Hongkun Hao, Zhiwei He, Yiming Ai, and Rui Wang. Improving open-ended text generation via adaptive decoding. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pages 62386–62404. PMLR, 21–27 Jul 2024a. URL https://proceedings.mlr.press/v235/zhu24d.html.

Yuqi Zhu, Jia Li, Ge Li, YunFei Zhao, Zhi Jin, and Hong Mei. Hot or cold? adaptive temperature sampling for code generation with large language models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 437–445, 2024b.