Methods and Applications for Probing Deep Neural Networks

by

Zining Zhu

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy

Department of Computer Science
University of Toronto

Methods and Applications for Probing Deep Neural Networks

Zining Zhu
Doctor of Philosophy
Department of Computer Science
University of Toronto
2024

Abstract

In recent years, the impressive abilities shown by deep neural network (DNN)-based systems have led to the curiosity towards the intrinsic mechanisms. The query towards these mechanisms has led to an emerging field, model interpretation. In this thesis, I focus on an approach for setting up model interpretation tests: probing. Probing refers to the approaches that use one or more small models to predict attributes from the representations of the larger DNN model. The probing test results lead to insights about the representations, which provide contexts to understand the behavior of the DNN models.

Probing has led to many intriguing findings about many types of DNN-based models, but their theoretical foundations are not complete. Two outstanding improvement areas are validity and reliability. The validity problem concerns the question of whether the measured results indeed match the terms we intend to measure. To quantify the validity, I apply tools from information theory to derive the error terms of several probing configurations. The reliability problem concerns whether the interpretation test results can be reproduced if this test were conducted again. I use tools in machine learning theory to derive a recommendation for setting up probing tests, requiring a suitable dataset size for conducting probing tests. This thesis also contributes to the utility of probing. I show that the probing results can be useful for predicting the downstream task performance of language models. I also show that probing results of the intermediate modules can lead to insights about the generalization performance. This thesis solidifies the methods and extends the applications for probing deep neural networks, including DNN language models and beyond.

Acknowledgements

I have received much help from numerous people, and the past few years are filled with gratitude. It's hard to enumerate since I only have limited space.

I would like to thank my supervisor, Frank Rudzicz, for teaching me various crucial components of doing research. You like to ask "so what?" — a question that I usually struggle to answer, but the thoughts and the discussions following this question usually lead to an "Aha" moment. The "Aha" moments are among the most important reasons I continue to pursue a career in academia.

I would like to thank other members of my supervisory committee: Yang Xu, Graeme Hirst, and Gerald Penn. I have also learned a lot from your feedback about the research ideas, drafts, and presentations. I would like to thank Colin Raffel for joining my committee as an internal-external member, and Anna Rumshisky for joining as an external member.

I would like to thank my colleagues at UofT, including in the SPOC lab, and other CL labs and beyond: Alex, Bai, Chuer, Demetres, Francois, Frank, Ian, Jade, Jinman, Jinny, Jixuan, John, KP, Lunjun, Michael, Mohamed, Raeid, Serena, Sheldon, Silviu, Soroosh, Toryn, Xindi, Yoona. The discussions with you have always been filled with interesting ideas. Due to obvious reasons, most of our discussions have taken place in a virtual space with four words starting with a "z", and our conversations usually start with a sentence testing for the audio quality. But we nevertheless witnessed (and perhaps have made some slight push towards) the progress of NLP in the past few years, in a closely connected community.

I would like to thank my colleagues at Amazon Search: my mentors Haoming, Jingfeng, Sreyashi, Chao, and other colleagues: Bing, Chen, Jiaxin, Jie, Jiri, Laurence, Peifeng, Qingyu, Rui, Tao, Xin, Yangqiu, Zhengyang, Zheng. The internship is filled with a ton of delicious foods, basketball games, and a first-hand look into how NLP technologies are deployed on commercial products.

I am fortunate to have been in some sort of mentor role with some collaborators: Esmat, Philipp, Rohan, Andrew, Jim, Jason, Huaizhi, Jingyu, Brandon. You have taught me a lot about both the research projects and beyond. I would like to thank you for tolerating my rambling especially when I'm brainstorming about the projects.

I would like to thank the instructors I have TA'ed for — Jonathan Rose, Albert Lai, Sean Robertson, Amir-massoud Farahmand, Demetres Kostas, and Michael Guerzhoy. You have been always looking for ways to let students learn and develop. The enthusiasm has inspired me profoundly.

I am fortunate to also have worked and/or discussed with many people, including Ana, Antoine, Chenhao, Edward, Francisco, Hanjie, Jekaterina, Jinlan, Malikeh, Marija, Reid, Sarah, Shumin, Veronica, Yanyang,

Yonatan, Xi, Xingyu, Zaiqiao, Zhiyuan. Your insights have been inspiring.

I would like to thank my friends: Bill, Haowei, Gordon, Zhongbin, among many others. The decade in Toronto would have been a lot paler without you.

I would like to thank my family for supporting my choices, many of them appeared absurd at that time. You have overcome great difficulties, yet still stand by my side. I would like to thank my fiancée, Nora. You have supported me through the ups and downs. I'm lucky to be with you. I would like to also thank the cats — Nunu, who likes to chime in for attention in video conferences, and Jinjin, who likes to hide in corners except when he wants sunlight or food — for their emotional support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

"In recent years, deep neural network have shown great capabilities." Many of my papers start with this line, just like many other papers published in recent years. This statement is usually followed by either the acknowledgment of an existing improvement area — "yet they still don't do well here" — or an inquiry towards the mechanisms — "so I would love to know why they perform so well". The former route leads to some approaches that improve the system's performance, and the latter route leads to some inquiry into the system itself. The latter inquiry is less popular but appears more interesting to me. This is the motivation for studying the problems in this thesis.

Why do the DNN-based systems perform so well across these tasks? This question is analogous to the procedure of "opening a black box", where the phase "black box" indicates the opaque properties of DNNs. The endeavor that tries to "open the black boxes" is also referred to as *model interpretability*. Model interpretation provides some contexts that allow more properties of the DNNs to be understood. The better-understood DNN models, together with the additional contexts, constitute a research direction that is usually referred to as *interpretable* AI.[1]

What does interpretable AI care about?

DNNs, from a structural point of view, are models containing parameters ("neurons"). Each neuron is trained to fit a simple goal (a transformation). A DNN can contain millions or even billions of parameters of this type. These parameters are organized by modules — weights, bias, attentions, representations, layers, and so on. One aspect that always makes me wonder is that, quoting LeCun et al. (2015), "With the composition of enough such transformations, very complex functions can be learned". An example of a complex

[1] A related term is explainable AI. While interpretable AI and explainable AI are frequently used interchangeably, they differ. Interpretable AI research aims at facilitating a better understanding of the systems' properties, whereas explainable AI research aims at revealing the functionalities of the systems. Broniatowski (2021) elaborated them in more detail.

function is the transformation of a movie review into a binary label that represents the sentiment polarity of the review (Maas et al., 2011).

Among all types of DNNs, I am mostly interested in DNN-based language models, or *pretrained* language models (PLMs). Loosely speaking, a language model computes the probability of a word given its context. This sounds like a simple goal, but if you concatenate the predicted new word to the context and repeat this process, language models can be used to generate long texts. Modern DNN-based language models like GPT-3 (Brown et al., 2020) can generate lengthy texts that resemble those written by humans, and many would argue that models of this type have passed the Turing's test for machine intelligence (Turing, 1950). In one of my recent works, we observe possibilities to use these language models to explain complex problems in styles that suit a variety of audiences (Zhu et al., 2023). The DNN-based language models demonstrate unique capabilities via language understanding and generation procedures.[2]

Language is widely used by humans to communicate thoughts and transmit knowledge. Both thoughts and knowledge are highly hierarchical. Perhaps not coincidentally, language also bears hierarchical structures at multiple levels: words, phrases, sentences, and paragraphs. A lot of works in the interpretable AI literature analyze these phenomena at each level, allowing us to design tests for a DNN-based model, both testing it as a whole or testing its component modules.

Tests, in my opinion, are at the center of interpretable AI.

A test specifies a subject and quantifies an ability of the subject. Usually, the subject is a component of a DNN model. The test results, together with the specifications of the testing protocol, contextualize various properties of the model. Let's take an example from Chapter 4 (which is originally proposed in Conneau and Kiela (2018)), bigram shift. The bigram shift test examines whether the language model can correctly recognize when the order of two words in a sentence is swapped. We can use a small, limited-capacity model to predict a binary label from the DNN model's representations at a certain layer. If this small model can easily separate the representations of the word-swapped sentence from the rest, we can infer that the language model, up to the specified layer, recognizes the sequences of the models well.

This small model is called a "probe"[3], and the approaches that use one probe, or a combination of probes, to study DNN models are referred to as "probing". There are many approaches to set up the tests to *interpret* neural networks. This thesis focuses on probing, which is arguably the most prevalent model

[2]It is still under debate whether the DNN-based language models "understand" language at all. I lean towards an empiricist's stance: if they perform well on a test, then we can claim they "understand" the part of language that is specified by the test. Note that the terms "perform well" and "specification" are ambiguous terms themselves. I defer to Kiela et al. (2021) and D'Amour et al. (2020a) for further discussions of the two concepts, respectively.

[3]Alternatively, the small model is also called a "probing classifier" if it is a classifier.

interpretation technique in the past few years. I will describe the foundations for probing methods in more detail in Chapter 2.

Probing is blessed by its flexibility and the benefits of the advance of machine learning research. It is a particularly fruitful approach. One of the representative works found that the DNN-based language models implicitly reconstruct a pipeline that resembles the traditional NLP pipeline: in general, the layers closer to the input process have more syntactic patterns, and the layers closer to the output process have more semantic patterns (Tenney et al., 2019a). Researchers have expanded the targets of probing to a wide range of abilities, including syntax and semantics (Jawahar et al., 2019; Kulmizev et al., 2020; Vulić et al., 2020), discourse (Koto et al., 2021; Zhu et al., 2020; Chen et al., 2019) and commonsense reasoning (Petroni et al., 2019; Lin et al., 2020).

Despite the wide application of probing, their theoretical foundations are not complete. Two prominent improvement areas are validity and reliability. Validity concerns whether the test results we acquire indeed reflect the true capability of the test subject. In Chapter 3, I describe a view from information theory. This view provides some intuitions to choose a suitable probe. Reliability concerns whether the test results can be reproduced by future examinations. In Chapter 4, I borrow some tools from machine learning theory and statistical power analysis. These tools allow us to recommend a suitable "data requirement". When there are more problems in a test than this requirement, we are fairly confident that a future tester will also get the same testing results (like "model A is better than model B on this task").

Now that we have the test results. A natural follow-up question is *So what?* In what ways does the gained understanding of the model's components benefit the developers of the AI models, in general? This is the question about the utility of probing, which I study in the later chapters. In Chapter 5, I describe how the test results of the pre-trained models can be beneficial for predicting how the models behave when we further fine-tune them. In Chapter 6, I apply probing to neural network models that involve images and show how the testing results at different modules of DNN models correlate to their generalization behavior.

Looking forward, the interpretation tests will be closely intertwined with the developments of the deep neural network models and will have more indications of how we should deploy these models. I elaborate several future directions in Chapter 7.

1.1 Thesis statement

The validity, reliability, and utility of probing can be improved by systematic studies. The utility has been studied widely, and I expand on the range of applications. The validity of probing remains unsolved; I

make an information-theoretic argument towards why using one single test (i.e., without a control test) is insufficient, and that carefully controlled tests are necessary. The reliability of probing, on the other hand, remains largely neglected. I scaffold a path towards quantifying the reliability of probing, using statistical learning theory.

1.2 Structure of this dissertation

- Chapter 2 describes some basics, defines the scope of probing, and reviews the related literature.
- Chapter 3 describes the validity problem of probing, and introduces some tools in information theory for studying the validity problem.
- Chapter 4 describes the reliability problem of probing, and introduces some tools in statistical learning theory for studying the reliability problem.
- The next two chapters, Chapters 5 and 6 describe the utility of probing, on LM and non-LM neural network models, respectively.
- Chapter 7 summarizes the works and presents some future directions.

Chapter 2

Foundations of probing

2.1 Neural networks

Machine learning models The neural network is a type of machine learning model. A model is a *mapping* machine. It maps the distribution of the input with the distribution of the output. Due to their high flexibility, models are considered *prediction* machines. Some examples of such prediction machines include:

- Given a sentence, predict an integer label that describes whether this sentence describes a positive sentiment.
- Given a sequence of words, predict the index of a word that is the most suitable to be concatenated at the end of the input sequence.

The first example is an instance of a *discriminative* model, where the model learns the difference between data belonging to different clusters. The second one can be considered as a *language model*, where the model learns the structures and various other aspects of languages. A language model is an instance of a *generative* model, where the model learns the data distribution.

Complex models A neural network contains multiple trainable parameters. These parameters are usually grouped in matrices. For example, one of the simplest forms of neural network models can be expressed as:

$$\mathbf{Y} = \sigma(\mathbf{W}\mathbf{X} + \mathbf{b}), \tag{2.1}$$

where $\mathbf{X} \in \mathbb{R}^{d_1}$ is the input, $\mathbb{Y} \in \mathbb{R}^{d_2}$ is the output. The matrix $\mathbf{W} \in \mathbb{R}^{d_2 \times d_1}$ is a weight, and $\mathbf{b} \in \mathbb{R}^{d_2}$ is the bias. The activation function $\sigma(\cdot)$ introduces some nonlinearity to the model. Some popular activation

functions include rectified linear unit (ReLU), sigmoid and softmax.

- Rectified linear unit (ReLU). $\sigma(z) = \max\{0, z\}$, where $z \in \mathbb{R}$. ReLU is an element-wise operation: when the input z has multiple dimensions, this operation is broadcasted.

- Sigmoid: $\sigma(z) = \frac{e^z}{1+e^z}$, where $z \in \mathbb{R}$. Sigmoid is also an element-wise operation. The output value of the sigmoid activation function is always between -1 and 1. Sigmoid has an appealing property: its derivative is $\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$.

- Softmax: For a d-dimensional input $\mathbf{z} = [z_1, z_2, ..., z_d]$, the output is also a d-dimensional vector, and its i^{th} dimension is: $\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\Sigma_{i=1}^{d} e^{z_i}}$. The softmax function can be considered as a multi-dimensional extension of the sigmoid activation function. For an input that spreads through the $(-\infty, \infty)$ spectrum, each value in the softmax's output is guaranteed to fall within $[0, 1]$. Additionally, the values of all dimensions sum up to exactly 1, so the softmax activation converts a distribution into a *probability* distribution.

The dimensions d_1 and d_2 are the cardinalities of the input and the output, respectively. Empirically, d_1 is the data dimension of the input data. On the other hand, d_2 equals the number of distinct labels that this model can predict. For a sentiment detector model that decides whether a sentence contains positive sentiments, $d_2 = 2$. For a language model that predicts the next word from a vocabulary with size $|V|$, $d_2 = |V|$.

Highly capable models The DNN models are highly capable, and the model developers are training increasingly capable models. There are many articles in conferences and journals (and more recently, in the newspapers) describing how capable they are. Here I will use a quantitative perspective to illustrate: the time to reach human-level performance on leaderboards, following Kiela et al. (2021).

The MNIST handwritten digit recognition dataset was released before 2000, and the performances of the best models were not comparable to human performances until over ten years later. Another dataset, SQuAD (Rajpurkar et al., 2016), was released in 2016. Merely 3 years later, the best models surpassed the human-level performances (Lan et al., 2020). Similar progress has been observed on a wide range of tasks, from speech recognition, text recognition to complex reasoning. Apparently, the DNN-based models have strong capabilities.

These strong capabilities arouse concerns from both the model developers and the model users. If we know how the intrinsics of the DNNs lead to their capabilities, we will be more equipped with knowledge

and tools to build better DNNs in the future. Also, the knowledge about how the DNNs work provide contexts in which the users can decide whether to trust the models, especially when using the AI models in high-stake scenarios. These concerns lead to an emerging research field, probing (or *model interpretation* in general).

2.2 Defining Probing

The term "probing" has been used in two related senses, a broad one and a narrow one.

The broader sense of "probing" is similar to that of "investigation" and "examine". In the broader sense, probing involves *any* data-driven methods that aim at investigating either the intrinsic or the behavior of the DNNs.

The narrower sense of "probing" imposes additional constraints on the methods applied to investigate the DNNs. In the narrower sense, probing specifically refers to the usage of one or more classifiers (or regressors) in a diagnostic manner, aiming at investigating the intrinsics or the behavior of the DNNs.

In the majority of the literature and the remaining part of this thesis, I adopt the narrow sense of probing. We can formalize probing as follows:

2.3 Probing configuration

Despite minor variations, a probing test consists of four common components: the task (probing dataset), the subject (model to be examined), the protocol (probing method), and the result (probing findings). Formally:

Definition 1 (Probing configuration)**.** A *probing configuration* $\mathcal{C}$ consists of $\{T, E, f\}$, where T specifies the probing task (e.g., past-vs-present from the SentEval suite (Conneau and Kiela, 2018)), E is the subject, usually an encoder that encodes the text specified by T into representation vectors (e.g., the output from the 11^{th} layer of BERT_base), and f is the probing classifier.

Remark. Task T can be a diagnostic dataset, for example, a text-based classification dataset with sequential inputs. The probe f can be either a classifier or regressor.

Each published paper on probing resembles the report of an exam or an investigation. Many of them use their own datasets, and each paper reports its own findings. My thesis largely focuses on the methodology. I aim to make the methods more valid, reliable, and useful. This section reviews some categories of approaches to set up probing tests, with a focus on the protocol.

Predicting features The earliest probing methods involved predicting engineered features from the representations. If it is easy to predict the feature from the representations, one can infer that the representation is highly correlative to the given feature. Gupta et al. (2015) used logistic regressors to recover features like the population of German from the word representations of the word "German". Ettinger et al. (2016) used classifiers to predict discrete labels from the representations of semantic composition models. Their methods (i.e., post-hoc investigation of the representations using classifiers/regressors) lay the foundation of probing methods.

Investigating DNN representations Several works extended the probing methods towards more complex models – more specifically, DNN-based models. Linzen et al. (2016) probed LSTM-based models for syntax-sensitive dependencies. Conneau et al. (2018) probed DNN-based encoders, including the skip-thought vectors (Kiros et al., 2015) and the multilayer LSTM encoders in machine translation models. Tenney et al. (2019b) probed a collection of models including BERT (Devlin et al., 2019), GPT (Radford et al., 2018) and CoVe (McCann et al., 2017).

These papers also compile diagnostic dataset suites that facilitate studies across multiple linguistic levels. Conneau et al. (2018) compiled the SentEval suite (Conneau and Kiela (2018) described more details), which has been used in multiple works in my thesis. The dataset suite of Tenney et al. (2019a) led to a finding that the DNN language models automatically form patterns that resemble the traditional "syntax-and-then-semantics" NLP pipeline.

Analyzing the geometry Predicting a feature from a representation can be interpreted as computing the correlation of a representation to a feature. Further, this approach can be framed as analyzing the geometry. This perspective leads to another framing of probing: reconstructing a distance from the representations. Hewitt and Manning (2019) attempted to reconstruct the syntactic distance from the representations. Relatedly, Niu et al. (2022) attempted to reconstruct the R-H distance (Roark and Hollingshead, 2008) from the representations. Probing by analyzing the geometry, among other interpretable AI methods, facilitates the understanding of the distribution patterns of the neural networks.

Measuring information Another viewpoint to contextualize the probing test is information estimation. Alain and Bengio (2017) first formulated this viewpoint, stating that the goal of probing is to examine whether the given representation contains the specified type of information. Pimentel et al. (2020c) further formalized the information estimation viewpoint. Following their frameworks, I proceed with studying the

validity of probing in Chapter 3.

Extensions of probes In recent years, researchers have proposed some probing methods without using diagnostic classifiers or regressors. Zhou and Srikumar (2021) used hierarchical clustering algorithms and computed the properties of the regions with given labels. Torroba Hennigen et al. (2020); Li et al. (2021) used Gaussian mixture models to study the geometry of the representations. These works draw correlations between the computed attributed properties and the probing targets and have led to unique insights about the DNN models.

Literature review I defer to Belinkov (2021) for a more comprehensive review of probing methods. Rogers et al. (2020) reviews the findings from probing BERT models. Relatedly, a recent article, Bowman (2023), presents some new findings from a behavioral perspective.

2.4 Putting probes together

Limiting each probing analysis to only one diagnostic prediction is unnecessary. The combination of multiple probes can lead to novel discovery. This section reviews some approaches in the literature.

Control task and amnesic probe Hewitt and Liang (2019) used a control task to contrast the original probing task. The control task was set up so that all information relevant to the measured target was removed. The performance difference between the original task and the control task was considered to quantify the informativeness of the representations. The construction of control tasks requires a deep understanding of the nature of the tasks. Elazar et al. (2021) proposed an approach to automate the constructions of control tasks. They derived algorithms that remove the principle component of the target from the representations. The processed representations were used to build the control tasks. I expand my discussions on this setup in Chapter 3. From an information-theoretic view (following the framework of Pimentel et al. (2020c)), the presence of control tasks improves the validity, while the probing goals are still imperfect.

Control tasks are useful for measuring the improvements beyond a given baseline (Hewitt et al., 2021). This approach is related to causal analysis, which is out of the scope of this thesis. I defer to two recent reviews: Feder et al. (2022) about causal analysis, and Kıcıman et al. (2023) about causal reasoning.

Probing across time Another approach is the combination across time. Liu et al. (2021a) set up probing tests across different steps throughout the training procedures. This combination allows the discovery of

the “speed” in which the DNNs acquire each type of linguistic knowledge. Further, some tasks involving complex reasoning, remain challenging throughout the training procedure of the RoBERTa models.

Chapter 3

An information-theoretic view for the validity of probing

This chapter is based on Zhu and Rudzicz (2020), published at EMNLP 2020.

3.1 Introduction

Let us start by studying the validity of probing, which concerns if we are measuring the intended quantity.

One of the earliest works in probing, Alain and Bengio (2017), provided an intuition about the intended quantity: probing queries the amount of information that is encoded in these neural network systems. Pimentel et al. (2020c) started to formulate this perspective using information theory. This chapter builds upon their information-theoretic framework, and studies whether the desired quantity matches the quantities that probing tests achieve.

In this chapter, we consider the diagnostic classifiers as probes (Alain and Bengio, 2017). These diagnostic classifiers are trained on pre-computed intermediate representations of neural NLP systems. The performances on tasks they are trained to predict are used to evaluate the richness of the linguistic representation in encoding the probed tasks. Such tasks include probing syntax (Hewitt and Manning, 2019; Lin et al., 2019; Tenney et al., 2019a), semantics (Yaghoobzadeh et al., 2019), discourse features (Chen et al., 2019; Liu et al., 2019a; Tenney et al., 2019b), and commonsense knowledge (Petroni et al., 2019; Poerner et al., 2019).

However, appropriate criteria for selecting a good probe are under debate. The traditional view that high-accuracy probes are better is challenged by Hewitt and Liang (2019), who proposed that the high accuracy

could be attributed to either (1) that the representation contains rich linguistic knowledge or (2) that the probe learns the task. To circumvent this ambiguity, they proposed to use the improvement of probing task performance against a control task (predicting random labels from the same representations), i.e., the "selectivity" criterion. Recently, Pimentel et al. (2020c), challenged this dichotomy from an information-theoretic viewpoint. They proposed to use an "information gain" criterion, which empirically is the reduction in cross entropy from a "control function task" probing from randomized representation.

In this chapter, we show the "non-exclusive-or" dichotomy raised by Hewitt and Liang (2019) is valid information theoretically. There is a difference between the original NLP model learning the task and the probe learning the task.

In addition, we show that the "selectivity" criterion and the "control function" criterion are equally valid. Pimentel et al. (2020c) formulated their errors with differences in a pair of KL divergences. We show that the error of the "selectivity" criterion (Hewitt and Liang, 2019) if measured from cross-entropy loss, can be formulated in the difference in a pair of KL divergences as well. When randomizations are perfect, these two criteria differ by only constant terms.

Empirically, on a POS tag probing task on English, French, and Spanish translations, we show that the "selectivity" and the "control function" criteria highly agree with each other. We rank experiments with over 10,000 different hyperparameter settings using these criteria. The Spearman correlation of the Hewitt and Liang (2019) vs. Pimentel et al. (2020c) criteria are on par with the correlations of "accuracy vs. cross-entropy loss" — two very strong baselines.

Overall, we recommend using control mechanisms to select probes, instead of relying on merely the probing task performance. When randomization is done well, controlling the target or representations is equivalent.

3.2 Information theory basics

Information theory provides a rich set of tools that help us process the *uncertainty* (Shannon, 1948). I will briefly review some concepts in this section.

Let us start by considering a random event of flipping a fair coin. Before a coin toss, it is equally likely that you will observe the head or the tail side, so there is a bit of randomness — we are uncertain about the result. After the coin toss, the knowledge of the observed result clears up this bit of uncertainty. How much information do we have when we observe the result? It is convenient to define this quantity as a *bit*: One bit of information clears up the uncertainty of a binary event.

The information is *additive*. When we toss two coins independently, knowing the result (which is one of the 2^2 scenarios) takes us 2 bits of information. Similarly, when we toss n coins independently, n bits of information help us determine the one scenario out of the 2^n equally likely ones.

In this problem, each scenario has probability $p = \frac{1}{2^n}$. We can describe this problem using a term, *random variable*, for example with symbol X. The viable space of this variable is noted with $\mathcal{X}$, and each observation of this variable is described by the *event*, x. Overall, there is a quantity that can describe the randomness of this problem: entropy H. The information theory literature defines H as:

$$H(X) = -\mathbb{E}_{x\sim\mathcal{X}}\log p(X = x). \tag{3.1}$$

The distribution of X follows a probability distribution $p(X)$. For simplicity, we can rewrite Equation 3.1 into:

$$H(X) = -\mathbb{E}_{p(X)}\log p(X), \tag{3.2}$$

and I will follow this notation in the rest of this chapter.

Let us now consider another random variable, Y that may be affected by X. When we observe X, we may observe some information about Y. This amount, the *mutual information*, equals the reduction of uncertainty of Y. Formally:

$$I(X;Y) = H(Y) - H(Y \mid X), \tag{3.3}$$

where $H(Y \mid X)$ is the *conditional entropy*, which describes the uncertainty of Y when the value of X is unknown.

3.3 An information-theoretic view for selecting probes

3.3.1 Formulation

We adopt the information-theoretic formulation of linguistic probes of (Pimentel et al., 2020c), and briefly summarize as follows.

We want to probe true labels T from representations R. An ideal probe should accurately report the code-target mutual information $I(T;R)$, which is unfortunately intractable. We will write $I(T;R)$ in an alternative form.

Let $p(T \mid R)$ be the unknown true conditional distribution, and a diagnostic probe, according to the setting in literature (Alain and Bengio, 2017; Hewitt and Manning, 2019; Hall Maudslay et al., 2020), is an

approximation $q_\theta(T \mid R)$ parameterized by θ, then:

$$\begin{aligned} I(T;R) &= H(T) - H(T \mid R) \\ H(T \mid R) &= -\mathbb{E}_{p(T \mid R)} \log p(T \mid R) \\ &= -\mathbb{E}_{p(T \mid R)} \log \frac{p(T \mid R) q_\theta(T \mid R)}{q_\theta(T \mid R)} \\ &= -\mathbb{E}_p \log q_\theta - \mathbb{E}_p \log \frac{p}{q_\theta} \\ &= H(p, q_\theta) - \mathrm{KL}(p \,\|\, q_\theta), \end{aligned} \tag{3.4}$$

where p and q_θ stand for $p(T \mid R)$ and $q_\theta(T \mid R)$ respectively, and KL stands for the Kullback-Leibler divergence. We also use $H(p, q_\theta) = -\mathbb{E}_p \log q_\theta$ to represent the cross entropy for simplicity.

3.3.2 The source of probing error

Traditionally, people use the cross entropy loss of the diagnostic probe $H(p, q_\theta)$ to approximate $I(T; R)$. We can derive a source of error by rewinding the above formulations:

$$H(p, q_\theta) = H(T) - I(T; R) + \mathrm{KL}(p \,\|\, q_\theta) \tag{3.5}$$

The first term on RHS, $H(T)$, is independent of either R or θ. Therefore, a low cross-entropy loss $H(p, q_\theta)$ can be caused by either of the two scenarios:

- High code-target mutual information $I(T; R)$, indicating the representation R contains rich information about the target T.
- Low KL-divergence between $p(T \mid R)$ and $q_\theta(T \mid R)$, indicating the probe learns the task.

The two scenarios exactly correspond to the dichotomy of Hewitt and Liang (2019).

3.3.3 The control task

In the following two sections, we describe two settings for applying controls. The control task, which is similar to Hewitt and Liang (2019), sets random targets for the probing task. Let us use $c(T)$ to indicate a "control function" applied on a token v that originally has label T. The control function could nullify the information of the input, except for what is associated with the word *type*.

If we measure the difference between cross-entropy in the control task and probing task

$$H(p(c(T) \mid R), q_{\theta_c}(c(T) \mid R)) - H(p(T \mid R), q_\theta(T \mid R)) \tag{3.6}$$

we can derive a form of error margin in their measurements[1]. Let us use a short-hand notations $H(p_c, q_{\theta_c}) - H(p, q_\theta)$ for clarity. Now, what does the difference between cross-entropy on the control task and the probing task contain?

$$\begin{aligned} & H(p_c, q_{\theta_c}) - H(p, q_\theta) \\ =& (H(c(T)) - I(c(T); R) + \mathrm{KL}(p_c \,\|\, q_{\theta_c})) - (H(T) - I(T; R) + \mathrm{KL}(p \,\|\, q_\theta)) \\ =& (H(c(T)) - H(T)) - I(c(T); R) + I(T; R) + (\mathrm{KL}(p_c \,\|\, q_{\theta_c}) - \mathrm{KL}(p \,\|\, q_\theta)) \end{aligned} \tag{3.7}$$

We already knew that $H(T) = \text{Const}$. According to the definition of the control function, the output $c(T)$ would be independent of R. Then:

$$\begin{aligned} & H(c(T)) = \text{Const} \\ & p(c(T), R) = p(c(T))p(R) \\ & I(c(T); R) = \mathbb{E}\frac{p(c(T), R)}{p(c(T))p(R)} = \text{Const} \end{aligned} \tag{3.8}$$

Therefore:

$$H(p_c, q_{\theta_c}) - H(p, q_\theta) = I(T; R) - \Delta_h \tag{3.9}$$

where Δ_h is a short-hand notation for the measurement error in the *control task* criteria:

$$\Delta_h = \mathrm{KL}(p \,\|\, q_\theta) - \mathrm{KL}(p_c \,\|\, q_{\theta_c}) + \text{Const} \tag{3.10}$$

When the probe fits the true distribution to a similar extent on both the control task and probing task, the error Δ_h would be small. Unfortunately, both KL terms are intractable.

[1]Note that Hewitt and Liang (2019) used accuracy instead of cross entropy. We discuss cross entropy to compare the errors against the control function (Pimentel et al., 2020c)

3.3.4 The control function

Control function (Pimentel et al., 2020c) introduces a random processor $\mathbf{c}(\cdot)$ on the representation R. To measure the information gain, they used an "information gain" criterion:

$$\mathcal{G}(T, R, \mathbf{c}) = I(T; R) - I(T; \mathbf{c}(R)) \tag{3.11}$$

Noticing that mutual information terms are intractable, they approximated the objective with the difference between cross-entropy in the control function task (we refer to as "control function" henceforth) and the probing task:

$$\tilde{\mathcal{G}}(T, R, \mathbf{c}) = H(p_c, q_{\phi_c}) - H(p, q_\phi) \tag{3.12}$$

To compute the error of this approximation, they reformulated the terms as follows:

$$\begin{aligned} &\mathcal{G}(T, R, \mathbf{c}) \\ &\quad = H(T) - H(T \mid R) - H(T) + H(T \mid \mathbf{c}(R)) \end{aligned} \tag{3.13}$$

The two $H(T)$ terms cancel out, then:

$$\begin{aligned} H(T \mid R) =& H(p(T \mid R), q_\phi(T \mid R)) - \mathrm{KL}(p(T \mid R) \,\|\, q_\phi(T \mid R)) \\ =& H(p, q_\phi) - \mathrm{KL}(p, q_\phi) \end{aligned} \tag{3.14}$$

$$\begin{aligned} H(T|\mathbf{c}(R)) =& H(p(T \mid \mathbf{c}(R)), q_{\phi_c}(T \mid \mathbf{c}(R))) - \mathrm{KL}(p(T \mid \mathbf{c}(R)) \,\|\, q_{\phi_c}(T \mid \mathbf{c}(R))) \\ =& H(p_c, q_{\phi_c}) - \mathrm{KL}(p_c \,\|\, q_{\phi_c}) \end{aligned} \tag{3.15}$$

Where we abbreviate similarly as we did for the control task. Specifically, we write ϕ for the probe parameters of the control function to tell apart from θ in the control task.

Pimentel et al. (2020c) showed that the error of their approximation, $\Delta_p = \mathcal{G}(T, R, \mathbf{c}) - \tilde{\mathcal{G}}(T, R, \mathbf{c})$, can be expressed as:

$$\Delta_p = \mathrm{KL}(p \,\|\, q_\phi) - \mathrm{KL}(p_c \,\|\, q_{\phi_c}) \tag{3.16}$$

Again, when the probe fits the true distribution to a similar extent on both the target labels distribution and the probing task, the Δ_p will be small. Unfortunately, both KL terms are intractable too.

3.4 Experiments

3.4.1 Setup

We use the same family of probes as Hewitt and Liang (2019) and Pimentel et al. (2020c), multiple layers perceptrons with ReLU activations, to show the correlations of their "good probe" criteria (control task and control function, respectively).

Overall, we sweep the probe model hyper-parameters with a unified training scheme on three tasks (probe, control task, and control function). The control task (function) setting includes labels (embeddings) drawn from a uniform random sample once before all experiments. In each training, we follow the setting of (Hewitt and Liang, 2019). We save the model with the best dev loss, report the test set loss and accuracy, and average across 4 different random seeds.

Data We use the Universal Dependency (Zeman et al., 2019) dataset loaded with the Flair toolkit (Akbik et al., 2018). We examine three languages: English, French, and Spanish. For the probing task, we use POS with labels provided by SpaCy[2]. We use the embedding of multilingual BERT (mBERT) implemented by huggingface (Wolf et al., 2020). If a word is split into multiple word pieces, we average its representations.

3.4.2 The "good probe" criteria largely agree

When measuring the qualities of probes using the "selectivity" (Hewitt and Liang, 2019) or "information gain" (Pimentel et al., 2020c) criterion, we show that the rules-of-thumb for training good probes largely agree.

- Early stopping before 24,000 gradient steps (approximately 4 epochs) could inhibit probe quality, but longer training procedures do not improve the probe qualities considerably.
- Smaller probes are better in general, but exceptions exist. For example, when weight decay is set to 0, probes with one hidden layer and 40 hidden neurons are better in both criteria.
- A small weight decay is beneficial.

We include more descriptions, including comprehensive experiment configurations and plots in Experiment Details (§3.4.4).

[2] `https://spacy.io`

Language	# POS	**# Tokens** train / dev / test	**Correlations** (**t_acc**,**f_ent**)	(**t_acc**,**t_ent**)	(**f_acc**,**f_ent**)
English	17	177k / 22k / 22k	0.1615	0.1334	0.1763
French	15	303k / 31k / 8k	0.0906	0.0606	0.1295
Spanish	16	341k / 33k / 11k	0.1360	0.0560	0.1254

Table 3.1: Spearman correlations between t_acc (the "selectivity" criterion (Hewitt and Liang, 2019)) and f_ent (the "gain" criterion (Pimentel et al., 2020c)) are on par with two "accuracy vs. cross entropy" correlations.

3.4.3 The two criteria are highly correlative

In addition to the qualitative correlations shown above, we compute the correlations between the two criteria over a grid-search style hyper-parameter sweep of over 10,000 configurations. For each "probe, control task, control function" experiment set, we record the following four criteria:

- **t_acc**: Difference between probing task and control **t**ask **acc**uracy. This is the "selectivity" criterion of Hewitt and Liang (2019).
- **f_ent**: Difference between control **f**unction and probing task cross **ent**ropy. This is the "gain" criterion of Pimentel et al. (2020c).
- **t_ent**: Difference between the control **t**ask and the probing task cross **ent**ropy.
- **f_acc**: Difference between the probing task and control **f**unction **acc**uracy.

We collect all experiments of each language according to these criteria, and use Spearman correlation to test three pairs of correlations. As is reported in Table 3.1, the (t_acc, f_ent) correlations are comparable to two strong baselines, (t_acc, t_ent) and (f_acc, f_ent), the correlations between measurements in accuracy and cross-entropy losses.

3.4.4 Experiment details and reproducibility

Hewitt and Liang (2019) proposed some rules-of-thumb to select good probes, including a "simple probe" suggestion. We sweep the hyperparameters to test whether these rules also apply when measuring probe qualities using the control function (Pimentel et al., 2020c).

Hyper-parameter ranges We sweep hyperparameters from the following ranges:

- Learning rate: $\{10^{-4}, 5 \times 10^{-5}, 3 \times 10^{-5}, 10^{-5}, 5 \times 10^{-6}, 3 \times 10^{-6}\}$
- Maximum gradient steps: $\{1500, 3000, 6000, 12000, 24000, 96000, \infty\}$. Their effects are shown in Figure 3.1.

- Weight decay: $\{0, 0.01, 0.1, 1.0\}$. Their effects are shown in Figures 3.2, 3.3 and 3.4. When sweeping weight decay, the max gradient step is set to 24000.

In any configuration mentioned above, we run four experiments with random seeds 73, 421, 9973, and 361091, and average the reported results (i.e., accuracy and loss).

Early stopping could inhibit probe quality Early stopping, if stopped before 24,000 gradient steps (approximately 4 epochs) may inhibit the quality of probes. In addition, we Figure 3.1 shows a high correlation between the "selectivity" (Hewitt and Liang, 2019) and "information gain" (Pimentel et al., 2020c) criteria.

Small weight decay is beneficial We find that smaller weight decays (e.g., 0.01) are more beneficial for probes than larger weight decays. While the two criteria rank the capacity of probes similarly, the most simple probes tend to stand out more distinctively with the "selectivity" criterion (Pimentel et al., 2020c), as are shown in Figures 3.2, 3.3, and 3.4.

Smaller probes are not necessarily better We find that while smaller probes have higher "selectivity" and "information gain" for mBERT representations, probes with one hidden layer and 40-80 hidden neurons are better than more simplistic probes, as shown in Figures 3.5, 3.6 and 3.7. The plots show consistency between the two criteria. For example, larger models and more layers do not necessarily lead to better results. Neither are the smallest probes with 0 hidden layers.

Note that we also swept hyperparameters for FastText, where probes with fewer parameters do not always outperform more complex probes in either accuracy, loss, selectivity, or information gain. Figures 3.8 and 3.9 illustrate these observations.

Reproducibility On a T4 GPU card, training one epoch takes around 20 seconds. Without setting maximum gradient steps, 98.6% of experiments finish within 400 epochs. We open-source our codes at `https://github.com/SPOClab-ca/InfoProbe`.

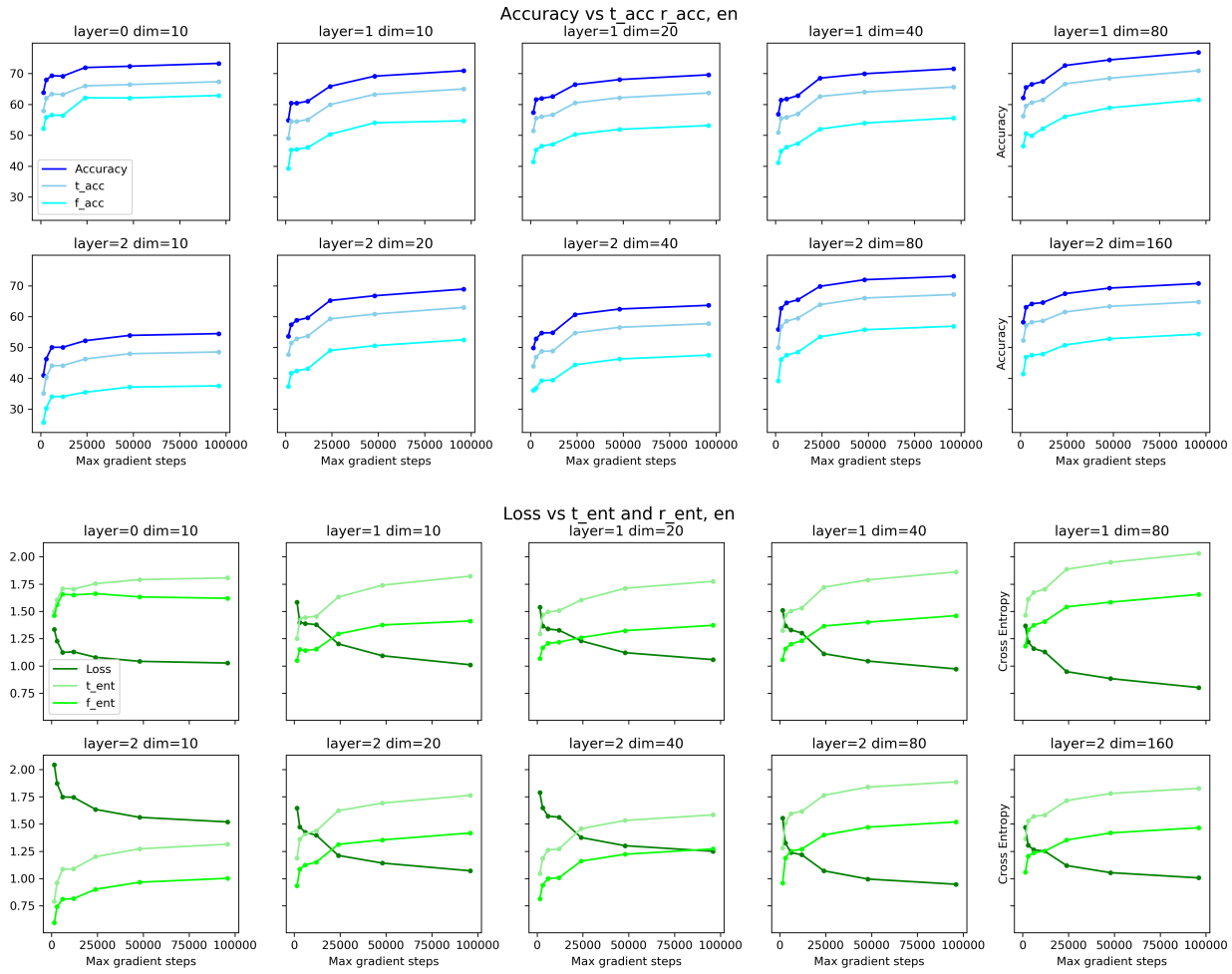


Figure 3.1: Max gradient step vs accuracy, t_acc, f_acc (blue) and loss, t_ent, f_ent (green) on English. The "t" refers to the control task (Hewitt and Liang, 2019), and "h" refers to the control function (Pimentel et al., 2020c). In this set of experiments, we look for the best learning rate and zero weight decay in each configuration.

3.5 Recent progress in the field

Improved information measurement Probing the DNNs requires measuring the relationship between high-dimensional variables. Directly estimating the mutual information is difficult, so one viable path is to compute some walkarounds. Zhu and Rudzicz (2020) fall into this avenue. One of the most impactful works along this avenue in recent years is predictive $\mathcal{V}$-information (Xu et al., 2020), which appeared in the same year as Zhu and Rudzicz (2020). Xu et al. (2020) used the difference between a pair of cross-entropy to describe the correlations between random variables. Subsequently, Pimentel and Cotterell (2021) formulated the difference of cross-entropy loss as Bayesian mutual information. Independently, Zhu et al. (2021) referred to the same term as task-specific information. While directly estimating the mutual information of high-dimensional variables has been hard, the avenue that develops information estimators has witnessed

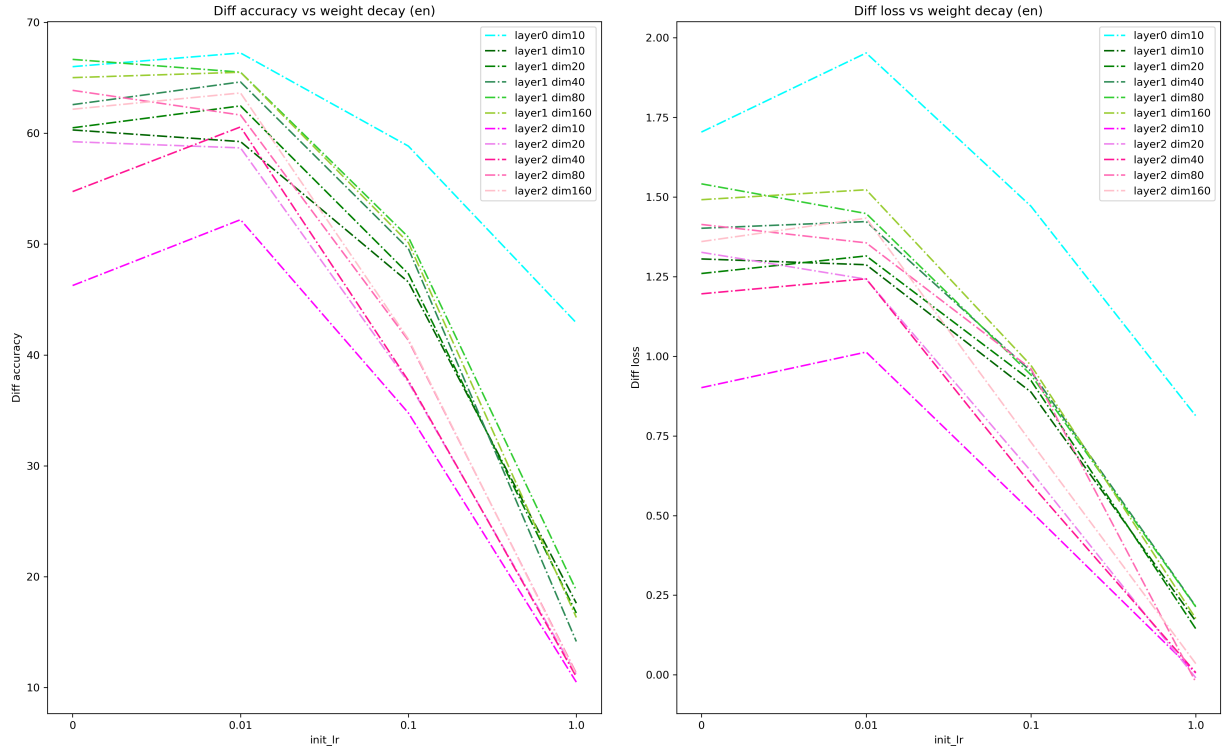


Figure 3.2: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria against weight decay on model configurations, on UD English. For each configuration, the learning rate leading to the highest accuracy is selected.

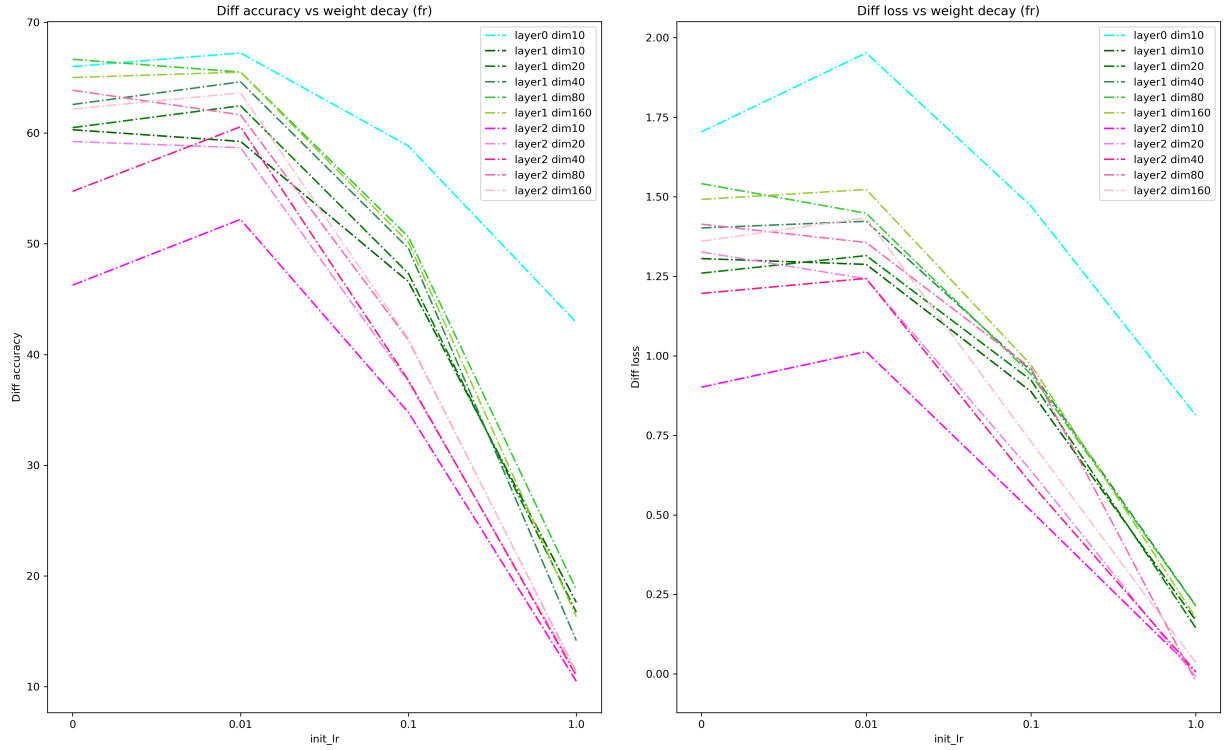


Figure 3.3: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria against weight decay on model configurations, on UD French. For each configuration, the learning rate leading to the highest accuracy is selected.

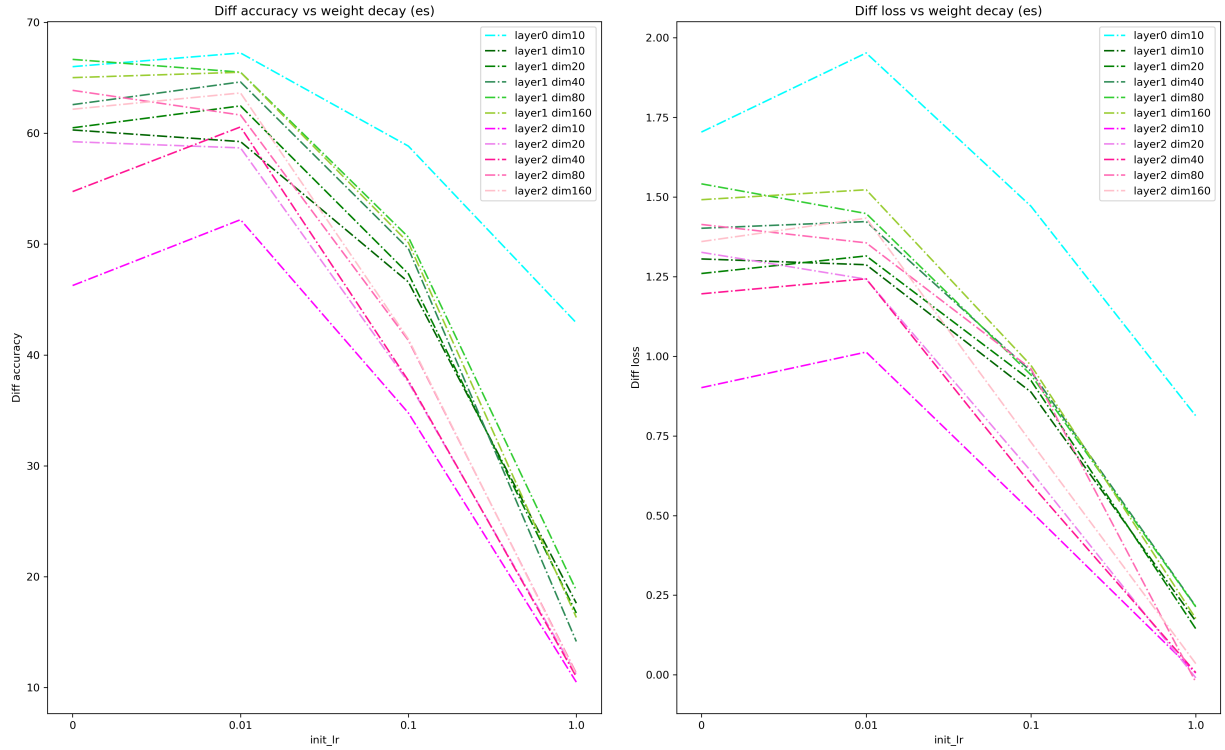


Figure 3.4: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria against weight decay on model configurations, on UD Spanish. For each configuration, the learning rate leading to the highest accuracy is selected.

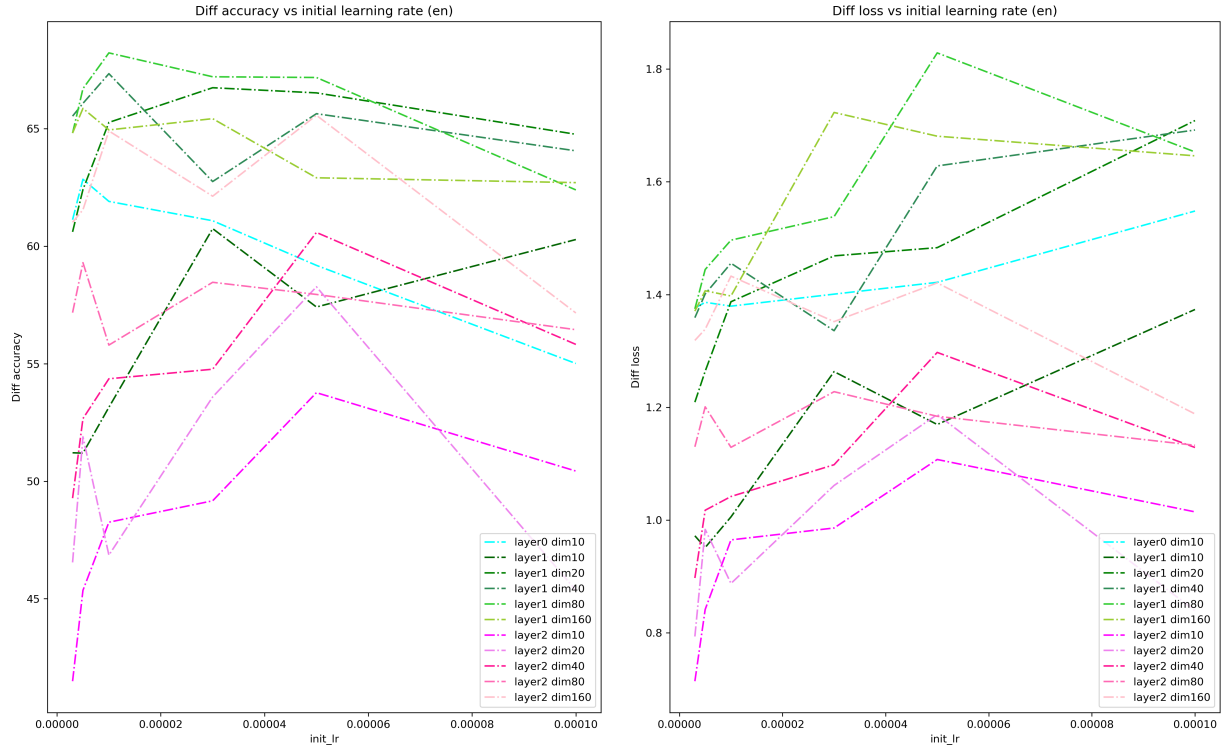


Figure 3.5: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria with different learning rates on model configurations, on UD English. The weight decay is set to 0.

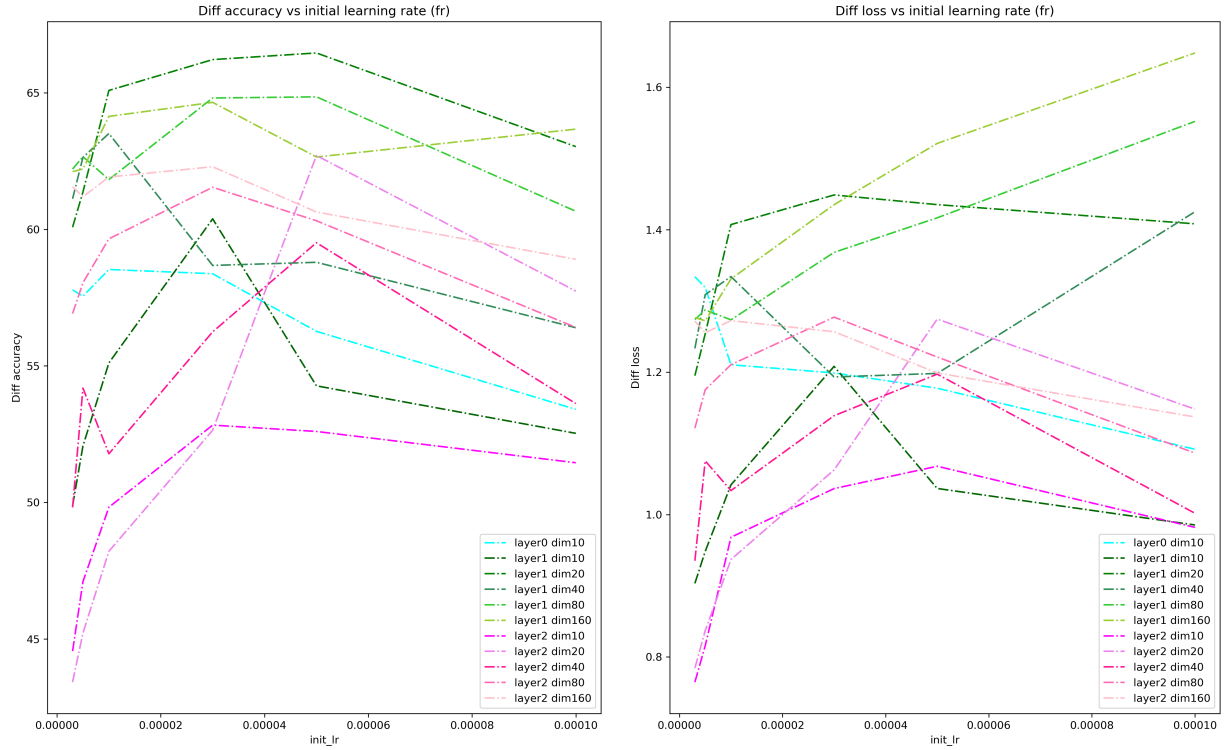


Figure 3.6: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria with different learning rates on model configurations, on UD French. The weight decay is set to 0.

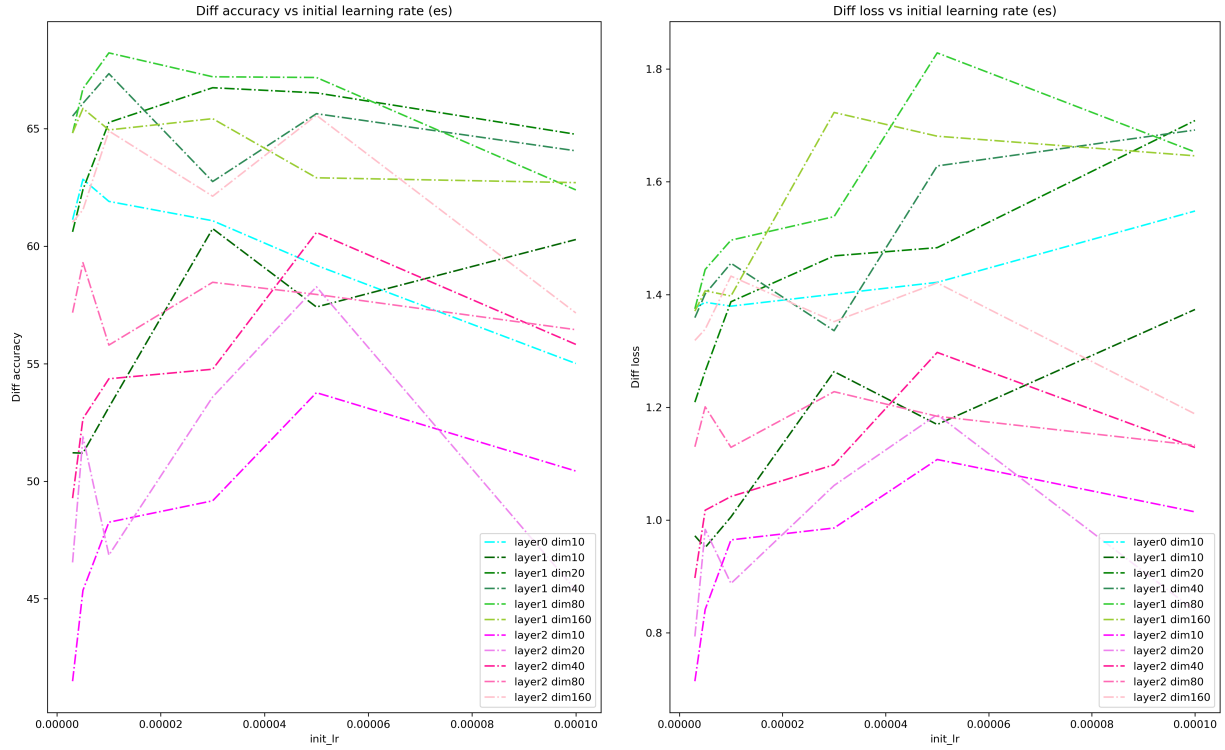


Figure 3.7: The "difference of accuracy" (Hewitt and Liang, 2019) and the "difference of loss" (Pimentel et al., 2020c) criteria with different learning rates on model configurations, on UD Spanish. The weight decay is set to 0.

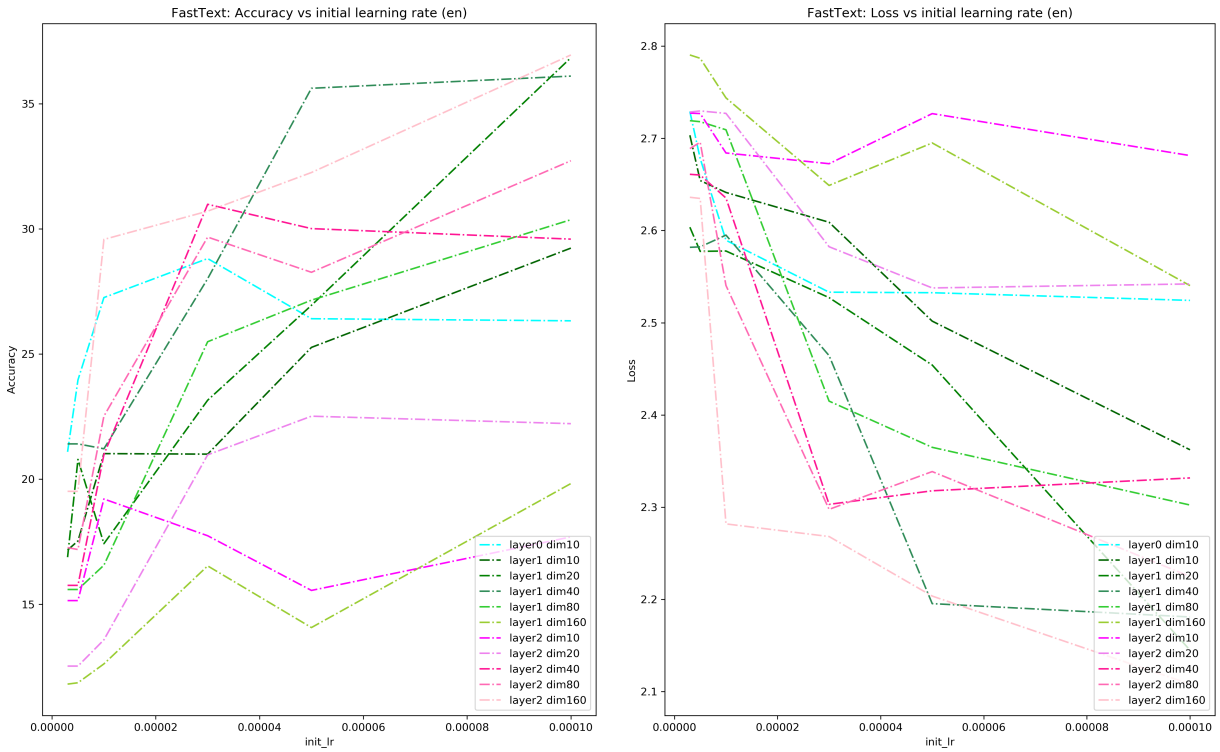


Figure 3.8: The accuracy and cross entropy loss of probes on FastText. These performances are much worse than those on mBERT, indicating the richness of information encoded in contextuality of mBERT.

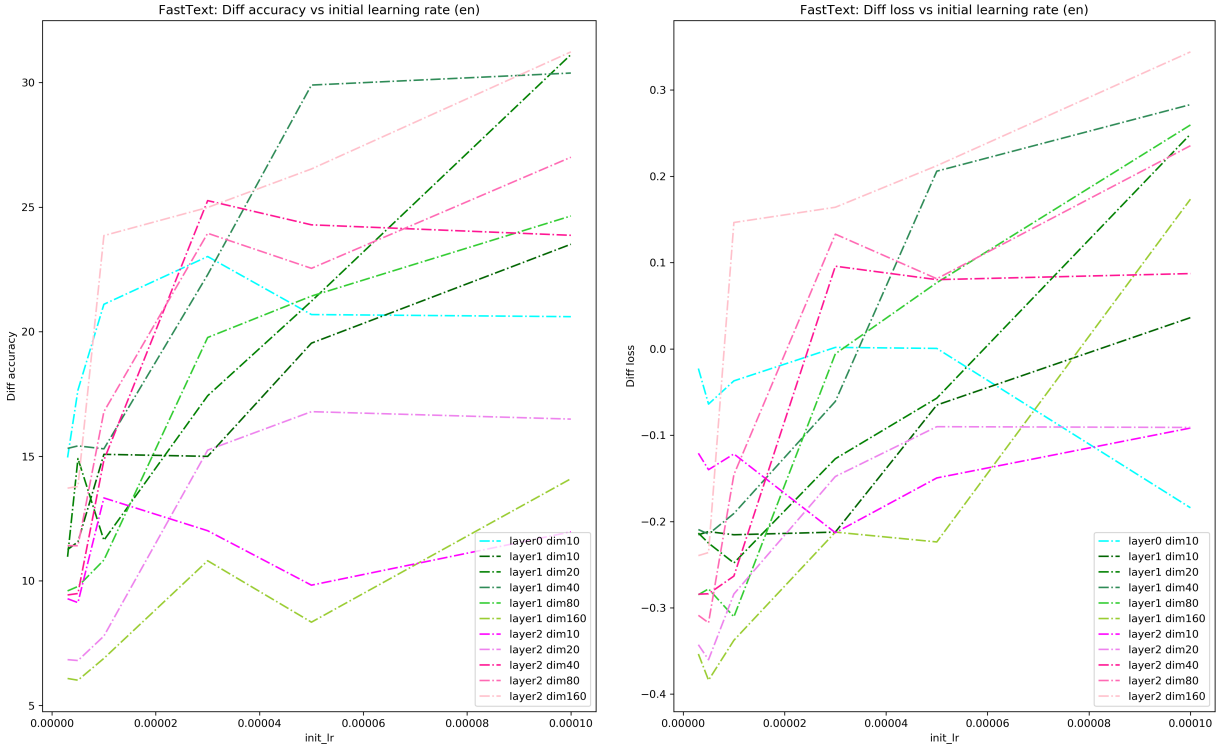


Figure 3.9: The selectivity (Hewitt and Liang, 2019) and information gain (Pimentel et al., 2020c) of probes on FastText. Probes with different capacities are ranked similarly using these two criteria.

significant developments. I defer a summary and an evaluation of recent information estimators to Czyż et al. (2023).

Alternative avenues towards validity The literature has several noteworthy approaches towards high-validity probes. I briefly summarize two here: minimum description length (MDL) (Voita and Titov, 2020) and Pareto hypervolume (Pimentel et al., 2020a). MDL formulates the probing problem as a procedure that transmits information from the source (i.e., model representation) to the target (i.e., the target) and uses the minimum description length to describe the quality of the representation. Pareto probing, on the other hand, considers a tradeoff between the probing performance and the complexity and calls for selecting the optimal diagnostic model that strikes a balance between the two factors.

3.6 Conclusion

This chapter describes a perspective of evaluating the validity of probing analysis following information theory. When selecting probes that better approximate $I(T;R)$, we recommend measuring *with a control mechanism* instead of relying on the traditional cross-entropy on probing task. We show both information-theoretically and empirically, that controlling the targets and representations are equivalent, as long as the control mechanism is randomized. While using control settings can reduce the impacts of the terms corresponding to "probe learning the task", the complete removal of these terms remains open.

Chapter 4

On the data requirements of probing DNNs

The contents in this chapter are based on the work of Zhu et al. (2022c), published at Findings of ACL 2022.

4.1 Introduction

This chapter considers the reliability of probing tests. If a probing test is reliable, we would expect the results to remain consistent when repeating this test. The reliability of testing is essential in quantitative studies, including medical, societal, and educational contexts (Kraemer, 1992; Drost, 2011; Golafshani, 2003). [1]

Depending on the forms of these tests, there are many ways to measure reliability. The test-retest reliability (usually quantified by Pearson correlations between the test and the retest results) measures the consistency of the results across time. The internal consistency reliability (usually quantified by Cronbach's α (Cronbach, 1951)) measures the consistency of participants responding to a set of items. The inter-rater reliability (quantified by Cohen's κ (Cohen, 1960) or Krippendorff's α (Krippendorff, 2018)) measures the extent of agreements between the annotators.

Existing works reflecting on the reliability of model diagnostics rely on repeating the tests on a variety of controlled conditions (Aribandi et al., 2021; Novikova, 2021). A larger variance in results indicates lower test reliability. In this chapter, I use tools in machine learning theory to translate a reliability criterion into a data requirement for the probing test data.

Probing papers in the literature are associated with datasets of varying sizes, as shown in Figure 4.1. What is a suitable size for probing datasets? Larger training datasets lead to tighter generalization bounds.

[1] The reliability is related to two other attributes — validity and robustness. *Validity* measures how well the test measures what it intends to measure. In a valid test, the result is right for the right reasons (McCoy et al., 2019; Ravichander et al., 2021). *Robustness* measures how well the results of a test can generalize from the experimental setting to real-world settings (Xing et al., 2020; Niu et al., 2020). This chapter focuses on reliability.

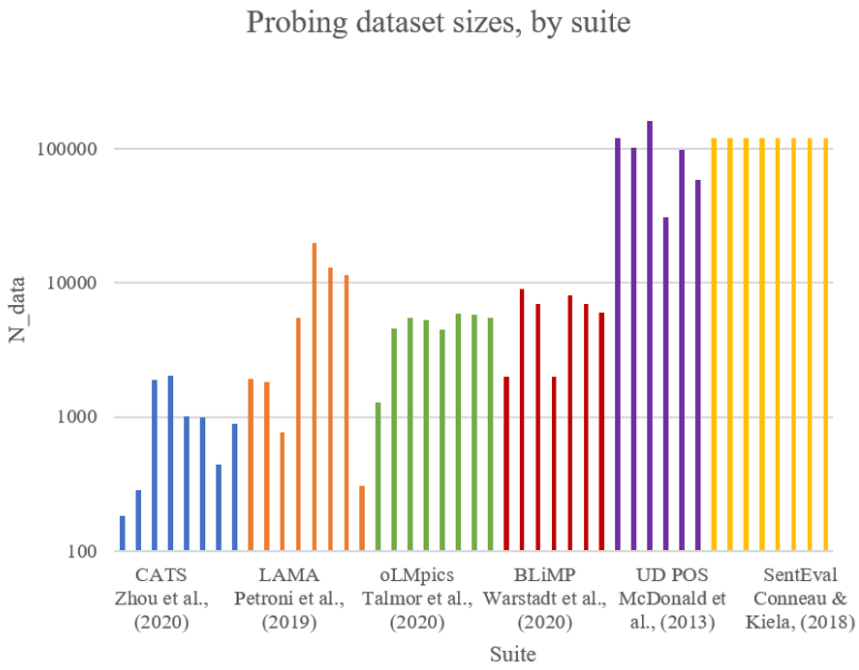


Figure 4.1: Sizes of the datasets in some common probing suites. Depending on the tasks, they vary from hundreds to larger than 10^5.

With other conditions fixed, larger testing datasets allow for higher statistical power in comparing probing models and classifiers. That said, it is neither realistic nor desirable to increase the probing dataset sizes arbitrarily. It is therefore essential to find a balanced size for probing classifications.

We propose a framework to estimate the data requirements of probing configurations (§4.2). Our framework considers the scenario of comparing probing configurations given some data. How many additional data samples may be necessary to reliably reproduce this comparison effect? We propose a novel method to estimate the required data samples by adapting a generalization bound.

We evaluate our framework on various probing tasks. First, we verify that the choice of generalization bound agrees with the probing results (§4.3.2). In a case study recognizing synthetic Gaussian noise, we verify that larger datasets provide higher statistical power (§4.3.3). We evaluate the utility of our framework on a collection of comparison problems (§4.3.4 - §4.3.5), where the probing data sizes recommended by the theoretical framework always supports power larger than 0.8. This chapter helps to formalize probing experiments. Our framework can be used by the research community in collecting probing datasets.

This chapter is related to the research avenues toward understanding the datasets. One way to study the dataset is to visualize each of the dataset samples on a map. Swayamdipta et al. (2020) mapped the data samples in NLP datasets to regions such as “hard-to-learn” and “easy-to-learn” using signals observed during the training process. Yauney and Mimno (2021) mapped the difficulty of data samples by the data-

dependent complexity (Arora et al., 2019). Vania et al. (2021) used Item Response Theory (Baker and Kim, 2004), a statistical framework from psychometrics to describe various attributes related to the difficulty of test set items. Similarly, researchers also describe the effects of datasets as a whole. Several papers attempted through information theory (Pimentel and Cotterell, 2021; Zhu et al., 2021). Le Scao and Rush (2021) compared the effectiveness of prompts to those of classification samples. When collecting datasets for probing, showing the effect of data is an important goal. Our framework considers the classification data samples, but we do not impose restrictions on the signal to probe in the classification tasks.

4.2 Methodology

4.2.1 Problem statement

Much of probing research considers some form of comparison problem. For example:

- Which deep neural language model encodes certain linguistic signals in an easier-to-extract manner?
- For a neural model, does pretraining with setting A support higher probing classification performances than models pretrained with setting B?
- Is the accuracy of a simple probing classifier (e.g., logistic regressor) higher than a more complicated one (e.g., MLP with hidden layers)?

These problems are instances of comparison problems, where we compare two probing configurations. Following the definition of probing configuration Def 1 (in Chapter 2), we can define the comparison as follows:

Definition 2 (Comparison problem)**.** A *comparison problem* consists of a pair of probing configurations, $\mathcal{C}_A = \{T, E_A, f_A\}$ and $\mathcal{C}_B = \{T, E_B, f_B\}$, where E_A and E_B are encoders, and f_A and f_B are probing classifiers.

Remark. Usually, the two configurations of a comparison problem, $\mathcal{C}_A$ and $\mathcal{C}_B$ differ by only one of $\{E, f\}$ to avoid confounds. A comparison problem can *collapse* if the two configurations are identical. Recommending the required number of data samples for a collapsed comparison problem is not meaningful.

4.2.2 An overview of the framework

For researchers collecting a probing dataset, we recommend the following procedure to estimate the probing dataset sizes:

1. Identify a comparison problem by specifying two probing configurations and collect a small set of data in a pilot study. Using the existing data, run the two probing classifications. Let R_1 and R_2 denote the probing performances, respectively.
2. When $|R_1 - R_2|$ is small, it is likely that the comparison problem collapses — §4.2.6 describes some heuristics to verify.
3. When the comparison problem has a difference in the performances $|R_1 - R_2|$, our framework can recommend the data requirements: Plug in $\frac{|R_1-R_2|}{2}$ to the generalization bounds to retrospectively solve for a recommendation of train data size N_{train}. §4.2.3 elaborates the generalization bounds in probing classifiers. §4.2.4 presents numerical examples.
4. Without loss of generality, we assume that the train, validation, and test data have relative sizes of $\eta : 1 : 1$. Then $(1 + \frac{2}{\eta})N_{\text{train}}$ is the total data requirement.

4.2.3 Generalization bounds

Machine learning theory literature provides many generalization bounds. These bounds are usually stated in the following form:

$$P\left(\left|R(\hat{f}) - R(f_*)\right| > B(n, \delta)\right) < \delta, \tag{4.1}$$

where f is the classifier, $R(\cdot)$ is the risk, n is the number of training data points, and δ is a hyper-parameter. Given n, a generalization bound states that, with a probability of at least $1 - \delta$, the risk of the empirically optimal classifier $R(\hat{f})$ differs from the risk of the globally optimal classifier $R(f_*)$ by at most $B(n; \delta)$. The risk is usually assumed to refer to the cross entropy loss. We show that several metrics used in probing have bounds with the form as well.

Accuracy The most widely used scores to measure the probing performance include accuracy, precision, recall, and F1 score. If we substitute the risk with accuracy, the bounds can apply without loss of generality, modulo two differences: accuracy (etc.) is bounded by $B = 1$ (whereas the upper bounds for loss could be larger), and is the highest with f_* (whereas the loss is the lowest with f_*).

Note that most behavioural probes[2] use evaluation metrics in this category. Many structural probes use additional evaluation metrics. We discuss them below.

[2] as described in Belinkov et al. (2020)

Control tasks It is possible that the probes, as diagnostic classifiers, rely on some irrelevant dataset statistics to boost the performance. To factor out this effect, Hewitt and Manning (2019) proposed to use control tasks. In a control task setting, we need to set up an auxiliary diagnostic classification task, and take the difference of the two classifications. Note that the difference is related to information-theoretic terms (Pimentel et al., 2020c; Zhu and Rudzicz, 2020). Regardless, their formulations involve some intractable terms that have to be empirically ignored.

Depending on the goal of the tasks, there are different ways to set up the auxiliary task. In the part-of-speech (PoS) probing task, for example, Hewitt and Manning (2019) associated each word type with a fixed PoS label. Another example is amnesic probing (Elazar et al., 2021), which uses iterative null-space projection (Ravfogel et al., 2020) to remove the probing task information from the representations.

Theorem 1. *The probing results for control tasks are subject to the generalization bounds in the following form:*

$$P\left(\left|R(\hat{f}) - R(f_*)\right| > 2B(n,\delta)\right) < \delta. \tag{4.2}$$

Proof. Let us use A_o and A_c to denote the original and the control task performance, and $\hat{f}$ and f_* (and $\hat{f}_c$ and $\hat{f}_{c*}$ for control task) to denote the empirical and the optimal classifier, respectively.

Since both A_o and A_c are count-based metrics, the analysis of the general form 4.1 gives us $|A_o(\hat{f}) - A_o(f_*)| \leq B_o$ and $|A_c(\hat{f}_c) - A_c(f_{c*})| \leq B_c$ with probabilities $1-\delta_o$ and $1-\delta_c$ respectively.

Then, with probability $(1-\delta_o)(1-\delta_c)$, we have $|A_o(\hat{f}) - A_c(\hat{f}_c) - (A_o(f_*) - A_c(f_{c*}))| \leq B_o + B_c$. In other words, a bound with the same form, hence the same convergence rate, as the count-based metrics still applies to the results of control tasks. □

Minimum description length Recently, Voita and Titov (2020) presented an alternative viewpoint of structural probing based on the minimum description length (MDL). The MDL of a probe or classifier is defined by the sum of (a) the code length required to transmit the data, and (b) the code length required to transmit the model for compressing the data. Voita and Titov (2020) gives two ways to approximate the MDL values: variational and prequential (i.e., online).

The *variational MDL* consists of two terms: the cross entropy loss $L(\hat{f})$, and the KL divergence between the posterior (β) and prior distribution (α) of the model parameters θ.

$$MDL_{\text{var}} = L(\hat{f}) + KL(\beta_\theta \parallel \alpha_\theta). \tag{4.3}$$

Theorem 2. *The probing results of variational MDLs are subject to the bounds taking the same form as Eq. 4.1.*

Proof. The generalization error of variational MDL is bounded by that of $R(\cdot)$, plus the estimation uncertainty of $\mathrm{KL}(\beta_\theta \parallel \alpha_\theta)$. In a Bayesian network implementation, the $\mathrm{KL}(\beta_\theta \parallel \alpha_\theta)$ can be acquired with less than 2×10^{-3} variance (Molchanov et al., 2017), bringing in a negligible additional uncertainty. In short, the generalization error of variational MDL is bounded by the cross entropy term. □

The *prequential/online MDL* (Voita and Titov, 2020) computes the code length required in this "transmission protocol". We first need to split the data into S shards, with their boundaries t_1, t_2, ..., t_{S-1}. First, transmit the first t_1 data points using random coding. The first shard (up to index t_1) constitutes of as few as 0.1% of the dataset. Then, optimize the model with the transmitted data, and transmit the next shard with the new model. Repeat this procedure until all the data are transmitted.

How to compute the prequential MDL? The first shard costs $t_1 \mathrm{log}K$ bits codelength (since each data point costs $\mathrm{log}K$ bits). The i^{th} portion constitutes of the points indexing from $t_i + 1$ to t_{i+1}, so the i^{th} shard costs $-\mathrm{log}p_{f_i}(y_{t_i+1..t_{i+1}}|x_{t_i+1..t_{i+1}})$ bits codelength. Putting the shards together, the total codelength required in this "transmission protocol" is:

$$\begin{aligned} MDL_{\text{pre}} &= t_1 \mathrm{log}K - \sum_{i=1}^{S-1} \mathrm{log}p_{f_i}(y_{t_i+1..t_{i+1}} \mid x_{t_i+1..t_{i+1}}) \\ &= t_1 \mathrm{log}K + \sum_{i=1}^{S-1} R(f_i; n_i). \end{aligned} \tag{4.4}$$

Theorem 3. *The generalization bound for prequential MDL takes the following form:*

$$P\left(\left|R(\hat{f}) - R(f_*)\right| > \frac{nC}{t_1} B(n, \delta)\right) < \delta, \tag{4.5}$$

where C is a constant.

Proof. Following likewise analysis, the generalization error of individual loss term is bounded by $\epsilon(n_i) = R(f_i; n_i) - R(f_{i*}) \leq B(n_i, \delta)$ with probability of at least $1 - \delta$.

Using a union bound, the error of summing up all these terms is bounded by the sum of all individual bounds. The error bound of prequential MDL is dominated by the first a few terms (i.e., the cross entropy losses with $n = \{0.1\%, 0.2\%, ...\}N$). □

Remark. Naturally, the theoretical bounds for prequential MDL appear "looser" than the bounds of previous metrics.

4.2.4 From generalization bounds to training data requirement

To estimate the required number of training data samples n, we can fix δ and enforce an upper bound on the excess risk $|R(\hat{f}) - R(f_*))|$. Then the corresponding n would be the required number of training data samples. Following is a numerical example where we consider the textbook finite function space bound[3]:

$$P\left(\left|R(\hat{f}) - R(f_*)\right| > B\sqrt{\frac{2\log\frac{2|\mathcal{F}|}{\delta}}{n}}\right) < \delta. \tag{4.6}$$

Here, we set $\delta = 10^{-8}$. How about the cardinality of the finite space? Following is a hand-wavey estimation. In a probing classification configuration $\mathcal{C} = \{T, E, f\}$, the encoder E produces vectors with $D = 768$ dimensions and f is a logistic regressor. Additionally, we assume that the $D+1$ weight parameters in f are stored in 32-bit floating point numbers[4], so each weight parameter takes 2^{32} possible values. Then the probing classifier constitutes a finite space with cardinality $|\mathcal{F}| = 2^{32} \times (D + 1)$.

When there are $n = 65,536$ training data points, the empirically optimal accuracy is different from the global minimum by at most 0.039 for $D = 4,096$ (InferSent) classifications with a probability of at least $1 - 10^{-8}$.

More importantly, we can also plug in an expectation on the generalization bound to retrospectively solve the training data requirement. For example, a bound of 0.05 requires $N = 40$k i.i.d data samples at $D = 4,096$.

If the datasets for both probing configurations in a probing classification are sufficiently large, the generalization bounds would be sufficiently small, so that the result of the comparison problem is reliable. As a heuristic, we let the bound be $\frac{|R_1-R_2|}{2}$, where R_1 and R_2 are the probing performances from the existing datasets. A tighter bound (e.g., $\frac{|R_1-R_2|}{10}$) requires more data samples (hence larger statistical power) as well as higher expectations for budgets. We consider $\frac{|R_1-R_2|}{2}$ to be a balanced choice.[5]

[3]This bound assumes that classifiers f come from a finite space $\mathcal{F}$. We defer to Theorem 7 of Liang (2016) for details. In the rest of this chapter, we use this finite-function-class bound.

[4]Following PyTorch's default for `FloatTensor`.

[5]Following are some justifications. In the most ideal case, both R_1 and R_2 are the true global minima $R(f_*)_1$ and $R(f_*)_2$, then the comparison results will remain consistent regardless of the number of data samples. In the less ideal case, both R_1 and R_2 are the empirical minima $R(\hat{f})_1$ and $R(\hat{f})_2$, then a generalization bound of $\frac{|R_1-R_2|}{2}$ indicates that the relative preference in the comparison will remain consistent (yet the scale of the comparison may fluctuate). We expect that most probing classifiers resemble this scenario since they reach almost perfect training accuracies. In the most unfortunate case, R_1 and R_2 deviate from the empirical minima. The extent they differ contributes to the randomness. While the scales of the empirical imperfectness remain unknown, one can consider some heuristics to reduce this imperfectness. First, a probing classifier with higher accuracy tends

4.2.5 Power analysis

We use power analysis to evaluate the reliability of our data recommendations. For a statistical test, the power is the probability of correctly rejecting the null hypothesis. In the context of This chapter, we compare the reliability of the prediction results provided by two probing configurations, $\mathcal{C}_A$ and $\mathcal{C}_B$. The hypothesis is stated as follows.

H_0: On a test set $\{x_i\}_{i=1}^{M}$, the results f_A and f_B are not significantly different.

To accept or reject H_0 on the two probing classifiers, we follow the approach of Card et al. (2020a) and apply a McNemar's test (McNemar, 1947), which checks if the χ^2 statistic is significant given the size of the test data. The χ^2 can be computed as $\chi^2 = \frac{(p_{01}-p_{10})^2}{p_{01}+p_{10}}$, where $p_{00}, p_{01}, p_{10}, p_{11}$ are the probabilities specified by the contingency table (Table 4.1).

	f_B incorrect	f_B correct
f_A incorrect	p_{00}	p_{01}
f_A correct	p_{10}	p_{11}

Table 4.1: Contingency table between two probing results, f_A and f_B.

Card et al. (2020b) described a framework that estimates the power by simulation. One repetitively samples a portion of test data and computes χ^2. The portion of simulations with significant χ^2 is taken as the estimated power. Empirically, one runs multiple classifications with distinct random seeds to increase robustness. To account for multiple classifications, we run the simulations of Card et al. (2020b) for each random seed, and then count the total number of significant simulations to compute the power. Usually, we expect that a reliable decision to reject the null hypothesis should have a statistical power of at least 0.8.

Throughout this chapter, we follow the rule-of-thumb assumption that the train set is four times as large as the test set. If the test set being large enough to support a 0.8 statistical power, the dataset size is large enough. In other words, the power analysis provides us with a way to *verify* the data size recommendations.

4.2.6 Detecting collapsed comparison problems

When we have data for a comparison problem from a "pilot study" and observe very small classification performance differences (e.g., of 0.5%), we might fall back to the null hypothesis — that the comparison problem collapses — in this case, increasing the data size does not "uncollapse" this comparison problem. Here we describe some heuristics to increase the confidence of detecting a collapse.

to have smaller empirical imperfectness, hence smaller unknown instability. Second, identifying the collapsed comparisons helps reduce the uncertainties introduced by the classifier imperfectness.

In our experiments, our data are subsampled from a larger dataset, so we can test if a probing configuration collapses by repeatedly subsampling the data, and run statistical tests. In the real-world, this is similar to running multiple "pilot studies" and collecting small-scale probing data, repeatedly. If the probing configurations output almost indistinguishable results, one can infer that the probing configuration collapses.

Alternatively, one can consider this augmentation method based on cross validation folds. For each dataset in a comparison problem, we divide it into, e.g., 6 folds. For each of $i = 1..6$ runs, take Fold i as the validation split, Fold $(i + 1)$ mod 6 as the test split, and the rest as the train split. Considering the probing classification results of all 6 runs can lead to higher confidence.

4.3 Experiments

4.3.1 Data and Models

Probing task We run probing classifiers on several classification tasks in one of the largest existing probing suites, SentEval (Conneau and Kiela, 2018): Past_present (tense prediction), bigram_shift (whether two words are flipped in a sentence), and coordination_inversion (whether two sentences are flipped) are binary classification tasks with 120k samples per class. Sentence_length contains 6 classes with 12k samples per class. To test the data requirements, we stratify sample subsets with $\{2^7, 2^9, 2^{11}, 2^{13}, 2^{15}\}$ training data samples per class, where applicable[6].

Encoders

- BERT (Devlin et al., 2019) is a contextualized language model. We take the multilingual $\text{BERT}_{\text{base}}$ model.
- SBERT (Reimers and Gurevych, 2019) encouraged semantically similar sentences to be mapped to nearby vectors in the representation space.
- GloVe (Pennington et al., 2014) is a static word embedding model. It maps each token to a fixed, 300-dimensional vector. We average all embedding vectors of a SentEval sequence as the input representation.
- InferSent (Conneau et al., 2017) is a contextualized language model. It processes the GloVe embeddings with a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) with 2,048 hidden dimensions.

[6]When there are 2^7 train sample per class, the subset consists of 256 train samples, 64 dev and 64 test samples, all with evenly distributed labels. i.e., $\eta = 4$. In this chapter, we keep this ratio consistent.

Probing classifiers We use a logistic regressor and a multilayer perceptron (MLP) with 20 hidden units (§4.3.6) as probing classifiers. In addition, we run several MDL probes, whose results are described in §4.3.11.

4.3.2 Verifying the theoretical bounds

We run probing classifications using a collection of subsets. Each subset is subsampled in a stratified manner from the dataset. We run 5 probing classifications with different random seeds on each subset.

To qualitatively examine the extent that the generalization bounds agree with the probing classifications, we plot both the empirical and the theoretical margins. Figure 4.2 shows an example. §4.3.7 contains additional plots. The empirical classification results reside within the theoretical margins, except for an outlier classification trial — this is the classification suboptimality, and we extend the discussion in §4.4.

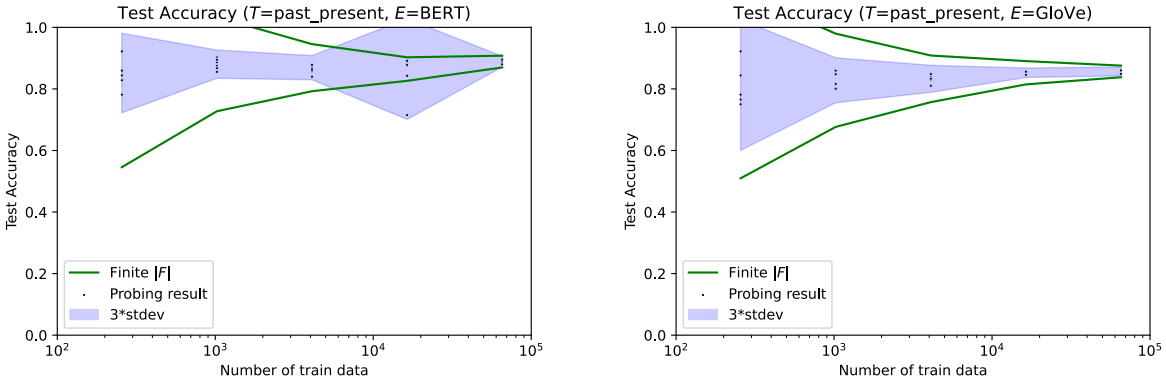


Figure 4.2: Theoretical bounds vs. empirical results on T = past_present, E = BERT (left) E = GloVe (right), and f = LogReg. The purple regions represent the empirical margin (mean $\pm$ stdev), while the green lines are the empirical mean $\pm$ margins computed by the learning theory bound.

4.3.3 Larger datasets support higher power

Intuitively, adding noise into the representation vectors makes it harder to decode the referential attribute. In this case study, we add Gaussian noise drawn from $\mathcal{N}(0, \sigma^2)$, where $\sigma^2 \in \{0.01, 0.03, 0.1, 0.3, 1, 3\}$, and compare against the probing classification with the original representations. Figure 4.3 shows the effect of noise on a configuration. §4.3.8 contains additional figures. Adding noise with a larger scale results in a configuration that is easier to distinguish. In addition, a larger training dataset usually leads to a higher power to distinguish the configurations.

This case study shows that we can verify the data requirement by incrementally collecting larger datasets for comparisons until we have sufficient power. For example, on the tense prediction task, distinguishing

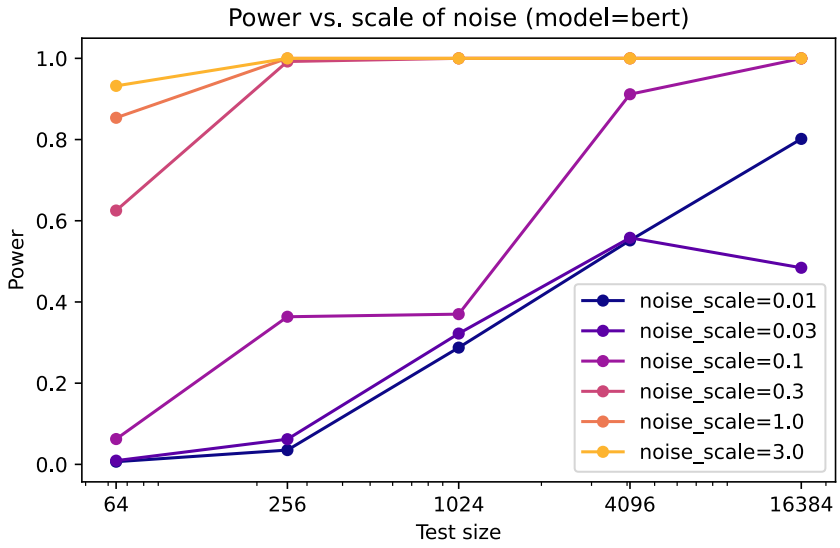


Figure 4.3: The powers to distinguish the representations with Gaussian noise from the original representations, for $T =$ past_present, $f =$ LogReg, and $E =$ BERT.

GloVe embeddings from its counterpart with $\mathcal{N}(0, 0.1)$ noise, $1,024$ testing data samples is sufficient to lead to 0.8 power. However, the same comparison with $\mathcal{N}(0, 0.03)$ noise requires up to $16,384$ testing data points.

The scale of Gaussian noise constitutes a spectrum. When we keep reducing the Gaussian scale, the comparison problem becomes more data-hungry. This leads to a question: where, on the "min-max" spectrum, do some other comparison settings (e.g., comparing between encoders) reside? In subsequent case studies, we verify that the numbers predicted by learning theory bounds have sufficient power.

4.3.4 Comparing to corrupted encoders

In this case study, we finetune the BERT models on WikiText[7] sentences with scrambled tokens for 200 steps.

Table 4.2 shows the recommended N_{train} values in the probing comparisons with corresponding "pilot data" (subsampled) sizes. As shown in Figure 4.4, the probing datasets with sizes no fewer than $N_{\text{test}} = 256$ (i.e., $N_{\text{train}} = 1024$) have sufficient power, and all recommended values fall within the "sufficient-power" range.

[7] `wikitext-2-v1` from huggingface datasets (Lhoest et al., 2021).

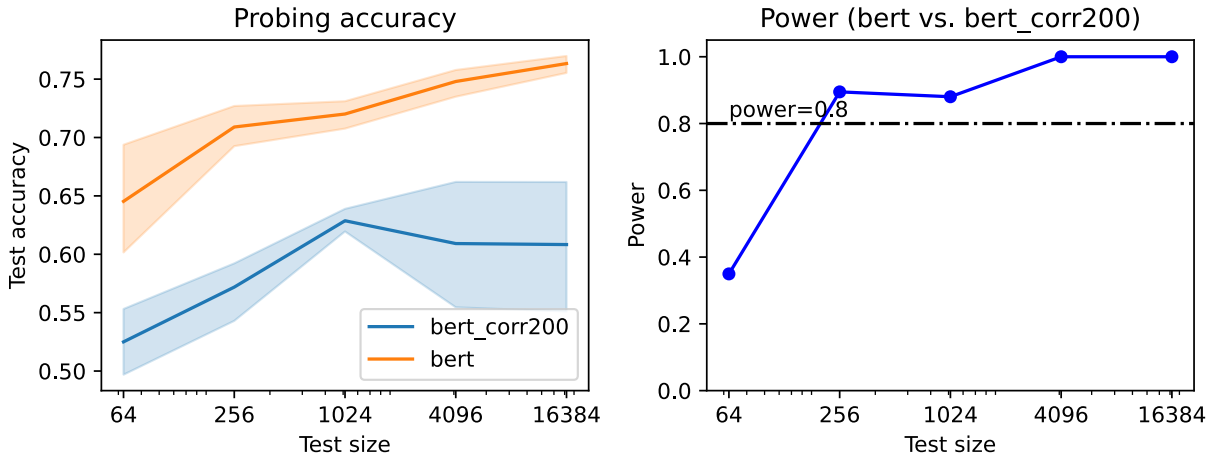


Figure 4.4: Left: the probing accuracies of BERT (orange) and BERT "corrupted" by 200 steps (blue). $T =$ bigram_shift, $f =$ LogReg. Right: the power to compare between them.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
64	0.1313	22,263
256	0.1281	23,362
1,024	0.0879	49,647
4,096	0.1331	21,662
16,384	0.1488	17,331

Table 4.2: The recommended N_{train} values in the comparison problem in Figure 4.4 given different subsample sizes.

4.3.5 Comparing between encoders

In this case study, we compare pairs of configurations containing the same task, data, and probing classifier but different encoders. Figure 4.5 shows an example. A test set of size $N_{\text{test}} = 1,024$ does not have sufficient power to compare the probing accuracy of BERT vs. GloVe, but $N_{\text{test}} = 4,096$ does. This corresponds to $N_{\text{train}=16,384}$, indicating that the recommendations in Table 4.3 are sufficient. §4.3.9 contains two other examples supporting the same finding.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
64	0.0344	324,563
256	0.0492	158,315
1,024	0.0355	303,516
4,096	0.0091	4,600,037
16,384	0.0320	373,513

Table 4.3: The recommended N_{train} values in the comparison problem in Figure 4.5 given different subsample sizes.

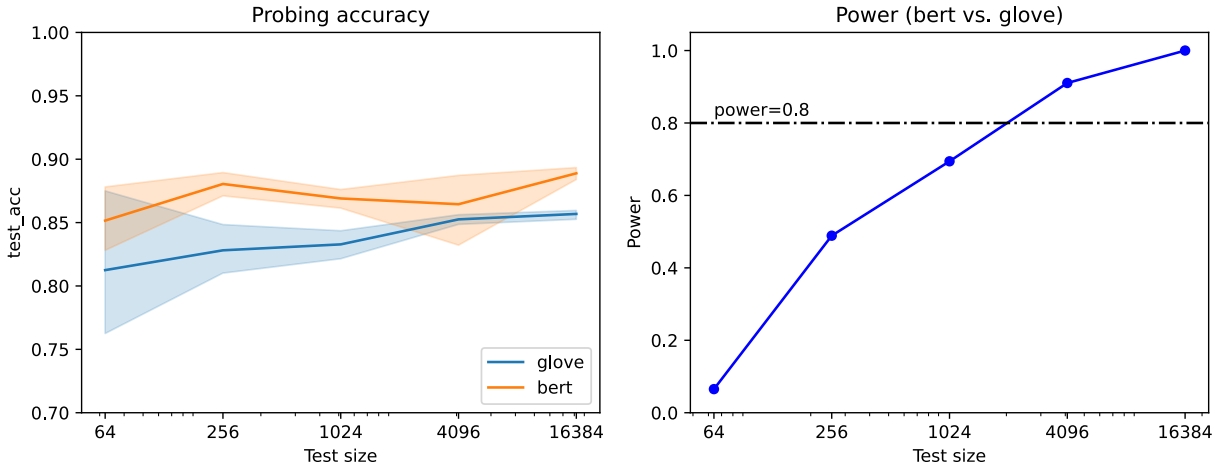


Figure 4.5: Left: the comparison of accuracies between BERT (orange) and GloVe (blue). T = past_present, f = LogReg. Right: the power of this comparison. The probing classification accuracy of BERT is higher than that of GloVe, but we do not have enough power to identify that until the testing dataset size is increased to $N_{\text{test}} = 4,096$.

4.3.6 Comparing between classifiers

Here, we compare two probing configurations with different classifiers: LogReg vs. MLP with one hidden layer of $H = 20$ neurons. Although the two configurations involve the same task, they have different training data requirements[8]. We take the larger one as the recommendation. Table 4.4 recommends N_{train} that are larger than the SentEval dataset sizes. These numbers are actually not necessary – Figure 4.6 shows that $N_{\text{test}} = 16.4$k (corresponding to $N_{\text{train}} = 65.6$k) is still insufficient to distinguish the results of the two probing classifier configurations on the bigram_shift task. There is insufficient evidence to reject the null hypothesis. In other words, the comparison problem between LogReg vs. MLP on T = bigram_shift collapses[9]. The exceedingly large data recommendations are meaningless.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
64	0.022	801,472
256	0.018	1,187,812
1,024	0.007	7,757,460
4,096	0.012	2,912,787
16,384	0.013	2,444,949

Table 4.4: The recommended N_{train} values in the comparison problem in Figure 4.6 given different subsample sizes.

[8]The LogReg has $D + 1 = 799$ parameters, while the MLP has $(D + 1)H + H + 1 = 15,401$, leading to a larger data requirement.

[9]Note that LogReg vs. MLP on other tasks e.g., T = sentence_length do not collapse. We include the results in §4.3.9.

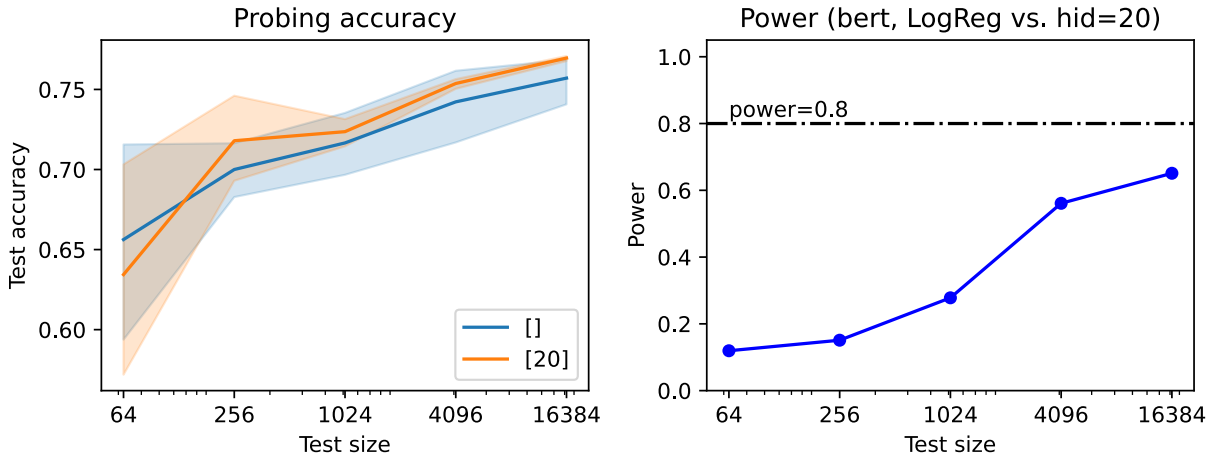


Figure 4.6: A comparison between probing classifiers. The task is T = bigram_shift. The encoders are both E = BERT. The probing classifiers are f =LogReg (blue) vs. f =MLP with 20 hidden neurons (orange), respectively.

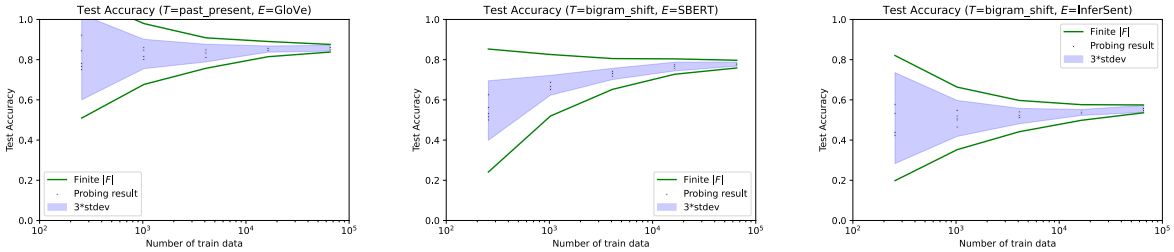


Figure 4.7: Theoretical bounds vs. empirical results, with f = LogReg. Left: T = past_present, E = GloVe. Middle: T = bigram_shift, E = SBERT. Right: T = bigram_shift, E = InferSent.

4.3.7 Theory bounds vs. experiment plots

We include additional theory vs. experiment plots in Figure 4.7. The purple regions represent the empirical margin (mean $\pm$ stdev), while the lines represent the margins predicted by the empirical mean $\pm$ the learning theory bound.

4.3.8 Power vs scale of noise plots

We include additional power vs. scale of noise plots in Figure 4.8.

4.3.9 Results on additional tasks

We list some results of additional tasks here:

- Table 4.5 and Figure 4.9 for T=sentence_length, BERT vs InferSent, f=LogReg.
- Table 4.6 and Figure 4.10 for T=coordination_ inversion, BERT vs InferSent, f=LogReg.

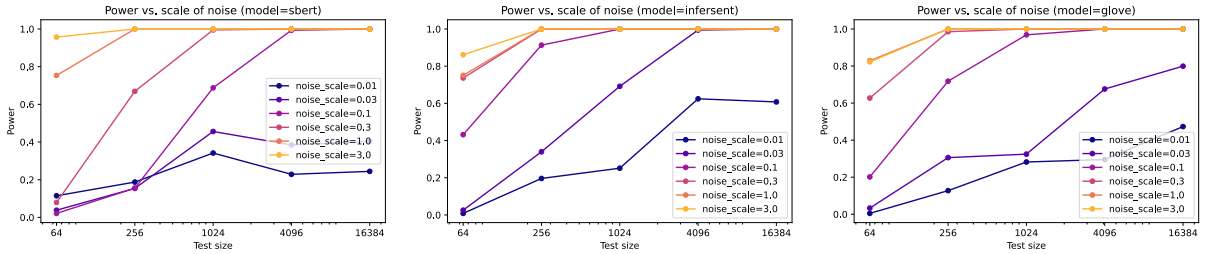


Figure 4.8: Additional power vs. scale of noise plots, on T = past_present, f = LogReg. Left: E = SBERT. Middle: E = InferSent. Right: E = GloVe.

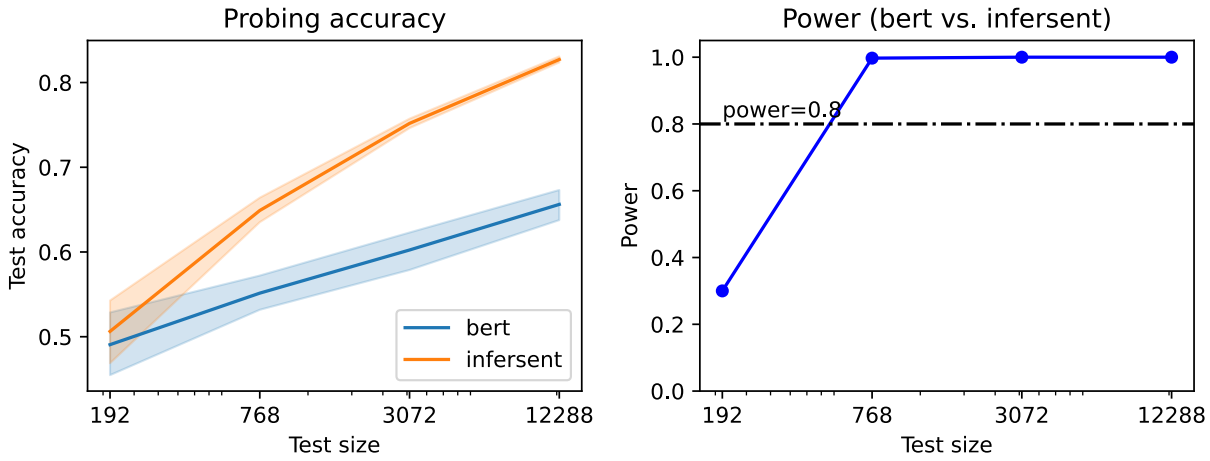


Figure 4.9: An example for T = sentence_length. Left: the probing performance of BERT (blue) and InferSent (orange). Right: the statistical power in this comparison.

- Table 4.7 and Figure 4.11 for T=sentence_length, E=BERT, LogReg vs MLP.

4.3.10 Hyperparameter configurations

We use Ray Tune to find the optimal hyperparameters for training. The search space include:

- Learning rate: 10^{-4}, 5×10^{-4}, 10^{-3}, 5×10^{-3}, 10^{-2}
- Batch size: 8, 16, 32, 64
- Number of epochs: we set it to 50. We stop running when the validation loss reaches a plateau for 5 epochs. Then, we report the result from the epoch with the highest validation accuracy.

We use pytorch to implement the models, and Adam (Kingma and Ba, 2015) to optimize. To reduce the training time, we cache the representation vectors. The runtime is about one minute per 200 training data points. Our analysis scripts are available at `https://github.com/SPOClab-ca/probing_dataset`.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
192	0.065	95,156
768	0.128	24,375
3,072	0.178	12,481
12,288	0.196	10,285

Table 4.5: The recommended N_{train} values in the comparison problem in Figure 4.9. Given different subsample sizes, the recommended N_{train} are greater than $3,072$ (i.e., $N_{\text{test}} > 768$). Their statistical powers are greater than 0.8.

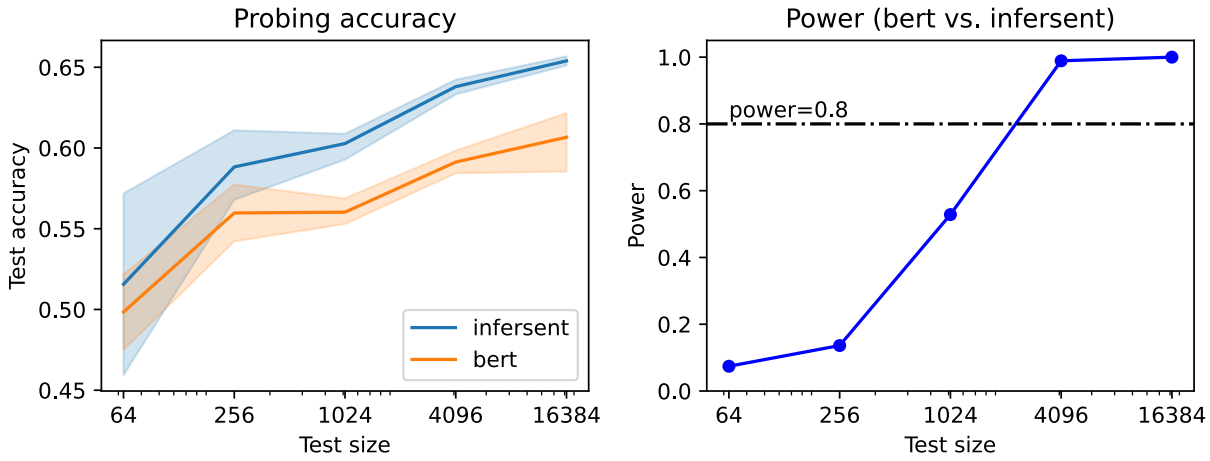


Figure 4.10: An example for $T =$ coordination_inversion. Left: the probing performance of BERT (orange) vs. InferSent (blue). Right: the statistical power in this comparison.

4.3.11 Other probing methods

To show the generalizability of our framework, we extend the experiments to two probing classifiers motivated by minimum description lengths: variational and prequential MDL probes (Voita and Titov, 2020). For the variational MDL probe, its results are affected by the arbitrary choice of prior (Pimentel et al., 2020b). Empirically, when we apply a uniform prior, the variational MDL usually degenerates[10], resulting in 0.5 accuracy. To alleviate this problem, varying of hidden layers and neurons in the probing classifiers is beneficial. For the prequential MDL probes, the results depend on the input sequence of data. Lovering et al. (2021) mentioned that the early steps sometimes produce cross-entropy losses that are larger than the uniform coding codelength. We also observe this effect, especially when the early steps contain imbalanced data.

[10]The classifiers output the same label for all inputs.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
64	0.025	635,040
256	0.019	1,128,961
1,024	0.041	231,499
4,096	0.051	151,861
16,384	0.063	100,153

Table 4.6: The recommended N_{train} values in the comparison problem of Figure 4.10. These values correspond to $N_{\text{train}} > 4096$, indicating statistical powers of greater than 0.8.

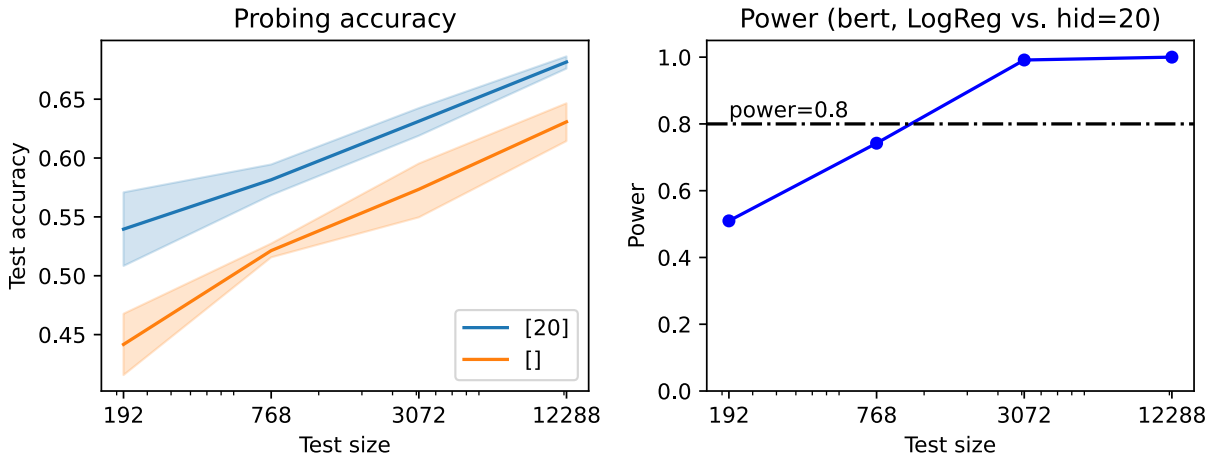


Figure 4.11: An example for T = sentence_length. Left: the probing performance of f=LogReg (orange) vs. f=MLP (blue). Right: the statistical power in this comparison.

4.4 Discussion

What does a high accuracy entail? Our framework implicitly considers the probing classification performances, but the causal relationship between, e.g., the accuracy, the data requirements, and the reliability can be explored further in future frameworks. A high probing accuracy indicates a small empirical risk $R(f)$. This could result from a small $|R(f) - R(\hat{f})|$ (the probe "learns the task"), or a small $R(\hat{f}) - R(f_*)$ (the distribution of the data samples represent the "true distribution" well). The two possibilities resemble the dichotomy raised by Hewitt and Liang (2019), but do they describe the same phenomenon? We leave this as an open question to future researchers.

On the stability of theoretical recommendations How stable are our theoretical recommendations? For those comparisons with sufficient evidence to reject the null hypotheses, the recommended N_{train} sometimes varies (e.g., at $N_{\text{test}} = 4096$ in Table 4.3). This is brought in by the suboptimality of several probing classifications. Future methods to estimate data requirement may improve the stability.

Subsampled N_{test}	Mean $\lvert R_1 - R_2 \rvert$	Recommended N_{train}
192	0.049	40,001
768	0.030	105,979
3,072	0.029	114,746
12,288	0.025	148,437

Table 4.7: The recommended N_{train} values in the comparison problem of Figure 4.11. The accuracies from MLP are higher than that of LogReg, but we do not have sufficient power until $N_{test} = 3,072$. This corresponds to $N_{train} = 12,288$, which the recommended N_{train} satisfy.

The suboptimality of probing classifier results The probing classifiers are usually imperfect. Due to the presence of, e.g., degenerative runs and local minima, the empirical result $R(f)$ may be different from the empirical optimum $R(\hat{f})$. While $R(f_*)$ describes the probing classification goal, the "easiness to extract", only $R(f)$ is empirically visible. As illustrated in Figure 4.12, the difference between the measured values $R(f)$ and the true global minimum $R(f_*)$ can be decomposed into two parts: $R(\hat{f}) - R(f_*)$, which is bounded by the generalization bounds, and $R(f) - R(\hat{f})$, which is the empirical imperfectness.

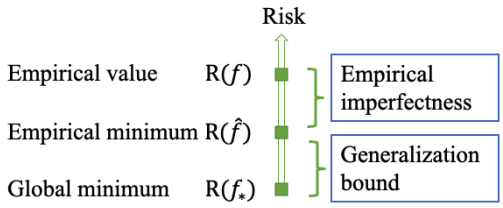


Figure 4.12: An illustration of the risk values.

Cross-task comparison problems Our framework does not consider cross-task comparison, i.e., when comparing $\mathcal{C}_A = \{T_A, E_A, f_A\}$ vs. $\mathcal{C}_B = \{T_B, E_B, f_B\}$ where $T_A \neq T_B$, because McNemar's test requires pairwise data. Alternative power tests would be necessary to consider cross-task comparison problems. We leave this to an open problem for future research.

Why not just collect as much data as possible? We argue in favor of knowing how many data samples we need, instead of directly collecting as many samples as budgets allow. The two views resemble the "top-down" vs. "bottom-up" research approaches mentioned in Bender and Koller (2020). Practically, our experiments show that many comparison problems do not need as many data samples as the sizes of some existing large probing datasets.

A "recipe" for probing datasets To systematically understand the abilities of neural networks, many more datasets need to be collected. Several recent papers desiderata for systematically constructing the

evaluation datasets (Ethayarajh and Jurafsky, 2020; Rodriguez et al., 2021), and this chapter follows the goal. To make the probing dataset collection procedure systematic, we consider a complete "recipe" beneficial. Our framework is one component of such a recipe, by quantifying questions of dataset sizes. Additional components for future work include quantifying the label distributions and the inherent "difficulties" of samples.

4.5 Conclusion

This chapter presents a novel framework to estimate the data requirements for probing experiments. This framework uses generalization bounds from formal learning theory to determine minimum training set sizes. In a series of comparison problems, we verify that our recommendations provide sufficient power. Our framework describes an actionable procedure to double-check if an experiment needs additional data samples to be scientifically meaningful. Additionally, we call for further attention to complete a systematic methodology in evaluating probing datasets and methods.

Chapter 5

Utility of probing from a performance prediction view

This chapter is based on Zhu et al. (2022b), published at EMNLP 2022.

5.1 Introduction

Previous chapters dealt with the validity and reliability of probing tests. Starting from this chapter, I will discuss another aspect of probing: utility. This chapter shows an aspect where the probing results can be useful, during the development of the DNN-based models.

Large-scale neural models have recently demonstrated state-of-the-art performance in a wide variety of tasks, including sentiment detection, paraphrase detection, linguistic acceptability, and entailment detection (Devlin et al., 2019; Radford et al., 2019; Peters et al., 2018). Developing systems for these tasks usually involves two stages: a pre-training stage, where the large neural models gain linguistic knowledge from weak supervision signals in massive corpora, and a fine-tuning stage, where the models acquire task-specific knowledge from labeled data. The fine-tuning results are widely used to benchmark the performances of neural models and refine the models' development procedures.

However, these fine-tuning results are summary statistics and do not paint the full picture of deep neural models (Ethayarajh and Jurafsky, 2020; Bender and Koller, 2020). As researchers are increasingly concerned about interpreting the intrinsic mechanisms of deep neural models, many data-driven assessment methods have been developed. These assessments usually follow the route of compiling a targeted dataset and running post-hoc analyses. Until now, one of the most popular interpretation methods is referred to as

probing. To probe a neural model, one uses a predictor to obtain the labels from the representations that are embedded using the neural model. Probing analyses on deep neural models revealed some low-dimensional syntactic structures (Hewitt and Manning, 2019), common-sense knowledge (Petroni et al., 2019) and (to some extent) human-like abilities, including being surprised upon witnessing linguistic irregularity (Li et al., 2021) and reasoning about space and time (Aroca-Ouellette et al., 2021).

From the viewpoint of data-driven assessments, both fine-tuning and probing can reveal the abilities of deep neural networks, but they appear to steer toward different directions:

In-domain vs. out-of-domain. Fine-tuning uses in-domain data — we evaluate the models on the same distributions as those in deployment. Probing, however, uses out-domain data: instead of simulating the deployment environment, the targeted datasets focus on diagnosing specific abilities.

Inclusive vs. specific. In fine-tuning, edge cases should be included, so the unexpected behavior after deployment can be minimized (Ribeiro et al., 2020) and the fine-tuning results can be stable (Zhang et al., 2021). On the contrary, the probing datasets are more specialized, so smaller datasets suffice.[1]

High performances vs. faithful interpretations. While fine-tuning methods are mainly studied from an algorithmic perspective to enhance the performance of language models, probing methods aim at assessing the faithfulness of language models. To fulfill the former objective, fine-tuning is accompanied by efforts in pre-training, collecting more data, building better representations, and exploring novel model architectures (He et al., 2021; Sun et al., 2021; Wang et al., 2021; Jiang et al., 2020). Conversely, the latter goal is pursued by borrowing inspirations from a variety of other sources, including psycholinguistic assessment protocols (Futrell et al., 2019; Li et al., 2022), information theory (Voita and Titov, 2020; Pimentel and Cotterell, 2021; Zhu and Rudzicz, 2020), and causal analysis (Slobodkin et al., 2021; Elazar et al., 2021).

In short, probing assessments are more specialized (therefore more flexible) and less computationally expensive. In contrast, the performance scores of fine-tuning assessments are more relevant to the design and training of deep neural models. **Can probing be used in the development of deep neural models?** This question involves two aspects:

- *Feasibility*: Are probing results relevant in the model development?
- *Operation*: How to set up probing analyses to get these useful results?

We attempt to answer both. For feasibility, we show that a crucial feedback signal in model development, the fine-tuning performance, can be predicted via probing results, indicating a positive answer to the

[1] Another viewpoint for the dataset requirement can be derived from learning theory. Loosely speaking, optimizing more parameters requires more data to reach stable results. Fine-tuning involves more parameters than probing. Zhu et al. (2022c) provides a more quantitative discussion.

feasibility question.

For operation, we run extensive ablation studies to simplify the probing configurations, leading to some heuristics to set up probing analyses. We start with a battery of probing tasks and evaluate the utilities both task-wise and layer-wise (§5.5.2 - §5.5.3). We then reduce the number of probing configurations, showing that as few as 3 configurations can predict fine-tuning results with RMSEs between 40% and 80% smaller than the control baseline (§5.5.5). To further answer the operation question, we run ablation studies on different probing configurations, including probing methods (§5.5.6) and the number of data samples (§5.5.7). We also analyze the uncertainty of the results (§5.5.8). Our analysis shows the possibility of using probing in developing high-performance deep neural models. All codes are open-sourced at `https://github.com/SPOClab-ca/performance_prediction`.

5.2 Related Work

Performance prediction Xia et al. (2020) proposed a framework that predicts task performance using a collection of features, including the hyperparameters of the model and the percentage of text overlap between the source and target datasets. Srinivasan et al. (2021) extended this framework into a multilingual setting. Ye et al. (2021b) considered the reliability of performance — an idea similar to that of Dodge et al. (2019). Our work differs from the performance prediction literature in the set of features — we use the probing results as features — and more importantly, we aim to show that the probing results can improve the interpretability in the development procedures of large models.

5.3 Predicting fine-tuning performance with probing

We present the overall analysis method and evaluation metric in this section. §5.5 elaborates the detailed experiment settings.

Predicting fine-tuning performance A deep neural model M can be fine-tuned on task t to achieve performance $\mathcal{A}_t$. Let $\mathbf{S} \in \mathbb{R}^N$ be the test accuracies of probing classifications on model M, using N configurations. For example, a deep neural model $M =$ RoBERTa can be fine-tuned to reach performance $\mathcal{A}_t = 0.85$ on a $t =$ RTE task. With post-hoc classifiers applied to the 12 layers of M, we can probe for 12 test accuracies on a probing task (e.g., detecting the past vs. present tense), which constitute of $\mathbf{S}$.[2]

[2]Following the default implementation of linear regression, we include an additional dimension in $\mathbf{S}^{(k)}$ to multiply with the bias term, so $\mathbf{S}^{(k)} \in \mathbb{R}^{N+1}$ in the following equations.

To find the pattern across a diverse category of models, we regress over K models (we will describe in §5.4.3). The collected probing results $\{\mathbf{S}^{(k)}\}_{k=1}^{K}$ can be used to predict the fine-tuning performance $\{\mathcal{A}_t^{(k)}\}_{k=1}^{K}$ via regression. Formally, this procedure optimizes for $N+1$ parameters, $\theta \in \mathbb{R}^{N+1}$ so that:

$$\theta_* = \text{argmin}_\theta \Sigma_k \left\| \theta^T \mathbf{S}^{(k)} - \mathcal{A}_t^{(k)} \right\|^2. \tag{5.1}$$

This procedure has closed-form solutions that are implemented in various scientific computation toolkits (e.g., `R` and `scipy`). The minimum reachable RMSE is, therefore:

$$\text{RMSE} = \sqrt{\frac{1}{K} \Sigma_k \left\| {\theta_*}^T \mathbf{S}^{(k)} - \mathcal{A}_t^{(k)} \right\|^2}. \tag{5.2}$$

RMSE-reduction While RMSE can evaluate the quality of this regression, it is insufficient for measuring the informativeness of $\mathbf{S}$ due to the discrepancy among the fine-tuning tasks t. Suppose we have two tasks, t_1 and t_2, where the probing results $\mathbf{S}$ can support high-precision regressions to RMSE $= 0.01$ on both tasks. However, on t_1, even features drawn from random distributions[3] might be sufficient to reach RMSE $= 0.02$, while on the more difficult task, t_2, random features could only reach RMSE $= 0.10$ maximum. The probing results $\mathbf{S}$ is more useful for t_2 than t_1, but RMSE itself does not capture this difference.

Considering this, we should further adjust against a baseline, the minimum reachable RMSE using random features.

$$\theta_{c*} = \text{argmin}_\theta \Sigma_k \left\| \theta^T \epsilon^{(k)} - \mathcal{A}_t^{(k)} \right\|^2, \tag{5.3}$$

where the random features ϵ are drawn from $\mathcal{N}(0, 0.1)$. Overall, the RMSE and the reduction from the baseline are computed as:

$$\text{RMSE}_c = \sqrt{\frac{1}{K} \Sigma_k \left\| \theta_{c*}^T \epsilon^{(k)} - \mathcal{A}_t^{(k)} \right\|^2}, \tag{5.4}$$

$$\text{RMSE\_reduction} = \frac{\text{RMSE}_c - \text{RMSE}}{\text{RMSE}_c} \times 100. \tag{5.5}$$

In the experiments, all RMSE and RMSE$_c$ values follow 5-fold cross-validation. We report the RMSE_reduction as the score that measures the utility of $\mathbf{S}$.

[3]Considering the small data sizes (i.e., the total number of models studied), even the "random features" drawn from random noises contain artefacts — patterns that can be used to regress the results.

5.4 Evaluation tasks and datasets

5.4.1 Fine-tuning tasks

We consider 6 binary classification tasks in GLUE (Wang et al., 2019) as fine-tuning tasks: `RTE` consists of a collection of challenges recognizing textual entailment. Given two sentences, the model decides whether one sentence entails the other. `COLA` (Warstadt et al., 2019) requires the model to determine if a sentence is linguistically acceptable. `MRPC` (Dolan and Brockett, 2005) requires the model to identify if a pair of sentences are paraphrases of each other. `SST2` (Socher et al., 2013) asks the model to output the sentiment positivity of movie reviews. `QNLI` contains questions and answers parsed from SQuAD (Rajpurkar et al., 2016). This task requires the model to decide whether the *answer* answers the *question*. `QQP`[4] tests if the model can correctly output whether a pair of Quora questions are synonymous.

5.4.2 Probing tasks

We use 7 probing tasks from SentEval (Conneau and Kiela, 2018) which can be approximately grouped in two categories, syntactic and semantic:

- Syntactic: bigram shift (BShift), and tree depth (TreeDepth)
- Semantic: past present (Tense), subject number (SubjNum), object number (ObjNum), semantic odd-man out (SOMO), and coordination inversion (CoordInv)

These probing tasks span across a range of linguistic abilities. In general, layers closer to the inputs (lower layers) in BERT contain more surface-level information, whereas higher layers contain more syntactic and semantic information (Tenney et al., 2019a; Jawahar et al., 2019), but the actual location of different linguistic features may vary (Miaschi et al., 2020). The SentEval datasets are usually hundreds of times larger than what would be sufficient to support statistically significant comparisons (Zhu et al., 2022c), so we randomly sample 1200 data points per class, corresponding to around 1% of the original SentEval data.

5.4.3 Pre-trained Language Models

We use several most widely used pre-trained language models for fine-tuning and probing. We refer to the models by their names on the Huggingface Model Hub.[5]

`roberta-base` (Liu et al., 2019b) pretrains BERT (Devlin et al., 2019) on over 160GB of English corpora, using improved techniques including dynamic masking, large mini-batches and masked language

[4] `https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`
[5] `https://huggingface.co/`

modeling without next-sentence-prediction.

`xlm-roberta-base` (Conneau et al., 2020) is pre-trained on 2.5TB of Common Crawl data from over 100 languages. The multiple languages sources improve the transferability across languages while compromising only a little accuracy on the English GLUE tasks (compared to the monolingual RoBERTa).

`albert-base-v2` (Lan et al., 2020) shares parameters across layers and decomposes the vocabulary matrices into smaller matrices. These parameter-reducing techniques reduce the computation resource requirements, which allows the model pretraining to further scale-up.

`microsoft/deberta-base` (He et al., 2021) uses separate attention vectors to model the content and the positions of each word. During fine-tuning, DeBERTa adds adversarial perturbations to the normalized embeddings.

`xlnet-base-cased` (Yang et al., 2019) models different permutation orders of the contexts during pre-training. XLNet additionally uses attentions to keep track of previous states, allowing the model to process the contexts extending beyond fixed lengths.

`Corrupted models`. To increase the diversity of models, we corrupt the language models on a masked language modeling task by MLM fine-tuning on scrambled Wikipedia[6] for 500, $1k$, $2k$, $4k$, and $6k$ steps. This "model augmentation" procedure does not apply to XLNet because scrambling the corpus produces a permutation of context, which XLNet already models. In total, there are 25 language models, each containing 12 layers.

5.4.4 Fine-tuning methods

For all fine-tuning classifications, we use the *AutoModelForSequenceClassification* framework by huggingface Transformers (Wolf et al., 2020). The model is trained with an AdamW optimizer with a collection of initial learning rates[7] and a batch size of 4. Since the GLUE tasks do not publicize the test set labels, we use the best dev set performance as the fine-tuning results. For reproducibility, we fix the random seed to 42 in PyTorch (Paszke et al., 2019). Additional details, including runtime and computation resources, are in Appendix 5.6.

5.4.5 Probing methods

There are many methods to probe a neural network. In this chapter, we use a post-hoc classifier to predict a target ("probing task target") from the representations of the first token (CLS). We run through a collection

[6] `wikitext-2-v1` from huggingface `datasets`.

[7] 1e-4, 3e-5, 1e-5, 3e-6, 1e-6, 3e-7. Note that most highest-performing classification runs are reached with either 1e-5 or 3e-6.

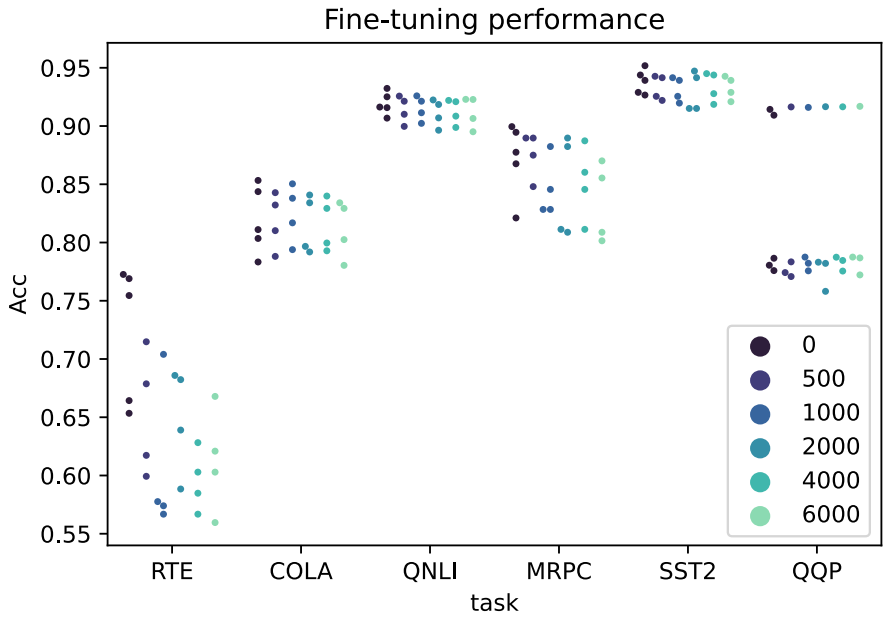


Figure 5.1: Fine-tuning performance. The color coding reflects the number of corruption steps on scrambled wikipedia, with 0 corresponding to the "vanilla" language models.

of scikit-learn (Pedregosa et al., 2011) classifiers,[8] choose the best one by the dev accuracy, and take its test accuracy as the probing result $\mathbf{S}$. Additional details, including runtime and computation resources, are in Appendix 5.6.

5.5 Experiments and Results

5.5.1 Fine-tuning performance

As an exploratory analysis, Figure 5.1 plots the distributions of the GLUE fine-tuning performances. The additional corruption steps result in more significant fine-tuning performance drops on RTE and MRPC than other tasks. Moreover, on QQP, the dev accuracies of `roberta-base` (and its corrupted models) are larger than 0.90, while most other models have around 0.80 dev accuracies.

5.5.2 Which probing task is most informative?

We start with testing the predictability of using the results from only one probing task. For each probing task, we concatenate the 12 probing results as features and predict the fine-tuning performance using linear

[8] Logistic Regression, MLP with 10 and 20 hidden nodes, Random Forest with 100 and 10 estimators, Decision Tree, and SVM. There are 7 classifiers in total.

	RTE	COLA	MRPC	SST2	QNLI	QQP	*Average*
All layers one task (§5.5.2)							
BShift	6.24	52.80	**53.18**	29.78	55.29	51.64	41.49
CoordInv	2.10	66.59	18.18	44.24	56.35	56.57	40.67
ObjNum	2.19	44.20	28.02	53.15	60.64	72.38	43.33
SOMO	30.90	44.75	29.39	29.28	38.64	55.68	38.11
Tense	3.07	48.42	34.65	22.29	41.37	75.58	37.56
SubjNum	-19.66	**78.56**	34.48	47.75	64.74	51.50	42.90
TreeDepth	4.37	53.03	9.54	46.98	62.79	54.67	38.56
One layer per task (§5.5.4)	36.12	62.66	25.78	49.87	59.79	26.73	43.49
Only three features (§5.5.5)	**41.69**	75.66	47.56	**72.59**	**80.52**	**76.77**	**65.80**
	CoordInv_1	ObjNum_2	TreeDepth_1	SubjNum_1	SubjNum_2	TreeDepth_6	N/A
	TreeDepth_1	SubjNum_2	SOMO_4	BShift_ 3	Tense_8	Tense_8	N/A
	BShift_12	TreeDepth_12	ObjNum_7	CoordInv_10	CoordInv_9	Tense_12	N/A

Table 5.1: RMSE reduction from baseline. A larger value shows the probing results more indicative of the fine-tuning performance. A small (or even negative) value means the probing results are not informative, compared to random features. The **bold-font** configurations are those with the highest RMSE reductions for predicting each fine-tuning task (i.e., within each column).

regression.[9]

Table 5.1 shows the percentage of RMSE reduction from baseline, using all layers from one probing task. There is no definitive answer towards "which probing task best predicts all fine-tuning tasks" but, depending on the linguistic abilities that each task targets, there are some regularities. For example, the 'number counting' probing tasks do not predict the fine-tuning performances on RTE, the textual entailment recognition task. In other fine-tuning tasks (COLA, QNLI, MRPC, SST2, QQP), however, each probing task shows positive RMSE reduction, signaling the ability to predict fine-tuning performance.

5.5.3 Which layers are the most indicative?

In the regression experiments of §5.5.2, we considered each feature equally important. However, a one-way ANOVA shows that some layers are more indicative than others, as Table 5.2 shows. For example, the probing results of tree_depth (at layer 1) and object_number (at layer 1) explain significant variance on all fine-tuning tasks.

Note that the layers with the most predictability should not be confused with those containing the richest linguistic knowledge. The former corresponds to the probing results that explain the most variances, while the latter corresponds with probing with the highest accuracy.

[9] `lm` method in the `caret` R package.

	RTE	COLA	MRPC	SST2	QNLI	QQP
bigram shift (BShift)	4,5	2,4,5	2,4,5,9	2,5,6	2,4,5	2,4,5
coordination inversion (CoordInv)	5,6,12	1,2,4,6	1,6	1,4,6	1,4-6	2-4,6
object number (ObjNumber)	1	1,3,8,11	1,3	1,3-5,8,11	1,3,8,11	1-5,12
semantic odd man out (SOMO)	4,5,8,12	2-6	3,4	3,5,6	2-6	2,5-9,12
past present (Tense)	1	1,3,5	1,5,6	1,11	1,3,5,8	1-5,8-11
subject number (SubjNum)	None	1,3-6,9	1	1,4	1	1,2,3,4
tree depth (TreeDepth)	1	1	1	1,3,5	1	1-3,7,8,11

Table 5.2: Layers with significant probing results ($p < .05$ from one-way ANOVA) with residual dof $= 12$.

5.5.4 Only one layer per probing task

Instead of probing all 12 layers, could using the probing results from only one layer for each probing task be beneficial? Following Table 5.2, we use the layers that are shown to explain significant portions of variance for the most fine-tuning tasks.[10]

The results are also included in Table 5.1. When reducing the number of features into around half (12 to 7), "one layer per probing task" can reduce more RMSE in RTE and SST2.[11] However, the results in other fine-tuning tasks indicate that alternative feature selection methods might help find a more predictive feature set.

5.5.5 Can we predict with only 3 features?

This experiment further reduces the number of features used while maximizing the MSE reductions. We iterate through all possible combinations of the $12 \times 7 = 84$ probing features for each fine-tuning task and report the largest RMSE reduction in predicting the fine-tuning performance.

Table 5.1 shows the results and the corresponding features. The prediction with three features can reduce the most RMSE on RTE, SST2, QNLI, and QQP. On COLA and MRPC, the RMSE reductions by the top three features are at most 6% smaller than those of the best previous configurations, which involved many more probing features.

The results show the utility of probing. It is possible to predict the fine-tuning performances by probing on as few as three configurations (each configuration using one probing task on one layer).

[10] Namely, layers 5, 6, 1, 5, 1, 1, 1 from the 7 probing tasks respectively.

[11] Note that the ANOVA in §5.5.3 use all data samples, so the choice of the features contain information propagated from the validation set. To ensure fair comparisons, we do *not* include the results from "one-layer-per-task" setting when finding the highest RMSE reductions in subsequent analysis. The results from this setting do not outperform the bold-font results even once, though.

	RTE	COLA	MRPC	SST2	QNLI	QQP	*Average*
Highest-accuracy probe in §5.5.2 - §5.5.5	41.69	78.56	53.18	**72.59**	80.52	76.77	67.22
Specify one probing method (§5.5.6)							
DecisionTree	51.98	68.48	54.31	70.90	74.35	52.85	62.15
LogReg	45.28	78.34	44.87	70.26	83.13	73.98	65.98
MLP-10	48.50	72.12	45.88	65.87	73.82	81.97	65.69
MLP-20	47.37	74.94	**63.79**	69.22	79.10	**82.67**	**69.52**
RandomForest-10	50.64	74.08	50.17	68.20	75.19	59.66	62.99
RandomForest-100	**53.94**	**79.20**	53.21	71.60	**83.25**	72.72	68.99
SVM	51.71	74.01	57.92	71.44	76.78	73.03	67.48

Table 5.3: Maximum RMSE reductions using different probing configurations. The **bold-font** numbers are the maximum values in each column.

5.5.6 Ablation: probing configuration

To further simplify the probing procedure, we run this ablation study. Instead of probing using a battery of post-hoc classifiers (as mentioned in §5.4.5), we test if the probing results from each individual classifier can reproduce the findings of §5.5.2 - §5.5.5.

Table 5.3 shows the maximum MSE reductions using different choices of probes. A perhaps surprising finding is that the probes selected from the "highest-accuracy" criterion do not always produce the most valuable results. To predict fine-tuning performances, directly specifying the probing method as MLP-20 or RandomForest-100 may be instead more recommended.

As a side note, among the 48 results presented in Table 5.3, only 9 are not achieved by the "best-3-features" methods (including the 2 shown in Table 5.1). This contrast emphasizes the importance of feature selection when configuring probes.

5.5.7 Ablation: dataset size

The findings in §5.5.2 - §5.5.6 show that as few as 1,200 samples per class (around 1% of total data) are sufficient to provide useful findings. What if we further reduce the sizes of probing datasets? Here, we repeat §5.5.2 and §5.5.5 with probing results from only 400 samples per class. While we can also reduce RMSE with only 400 samples per class, probing results are generally not as useful as those from 1,200 samples. Among the 48 configurations, the probing results from 400 samples have worse RMSE reductions in 11 configurations, but better in 5. Detailed results are included in Table 5.4.

	RTE	COLA	MRPC	SST2	QNLI	QQP	*Average*
All layers one task							
BShift	21.00	53.66	19.65	35.60	51.32	57.55	39.80
CoordInv	4.49	31.28	22.83	38.93	6.30	25.51	21.56
ObjNum	29.51	56.10	39.94	65.95	72.30	70.54	55.72
SOMO	-5.09	-7.14	16.09	9.36	-0.71	46.36	9.81
Tense	1.30	51.79	33.63	13.52	49.43	73.01	37.11
SubjNum	9.89	76.32	47.19	48.62	65.36	49.42	49.47
TreeDepth	-11.49	66.06	28.98	27.13	59.93	43.10	35.62
Only three features	47.38	77.84	56.70	72.27	82.01	71.08	67.88
	Tense_1	SubjNum_1	BShift_2	SubjNum_1	SubjNum_1	Tense_3	N/A
	SubjNum_11	BShift_6	ObjNum_7	Tense_2	SubjNum_8	BShift_4	N/A
	CoordInv_12	TreeDepth_8	SOMO_9	CoordInv_6	BShift_9	BShift_8	N/A

Table 5.4: RMSE reduction from baseline, using probing results with 400 data samples per class. The colored results are different from (better than or worse than) the results with 1,200 data samples (Table 5.1) by more than the estimated uncertainty margins in §5.5.8, i.e., 5% and 15% for 3 and 12 features, respectively.

5.5.8 Uncertainty analysis

Our method involves comparing the maximum RMSE reductions against the baseline (regressing from features drawn from Gaussian) RMSE_c, which may be affected by the random seeds. Here we describe an error analysis on the baseline regressor results of §5.5.2 - §5.5.5.

We run $N = 100$ Monte Carlo simulations on each configuration of regression from 3, 7, and 12 features, respectively, record the RMSE_c, and analyze the uncertainty. We use the variation of RMSE_c (as measured by $\text{Std}(\text{RMSE}_c)$) relative to the scale (as measured by $\text{Mean}(\text{RMSE}_c)$) to describe the uncertainty. As shown in Table 5.5, the uncertainty remains relatively stable across the choice of regression tasks but increases with the number of features. This result favors the use of fewer probing results as features.

Note that these uncertainty values are nontrivial. Let us take COLA as an example. To regress the fine-tuning performance, a 3-feature setting can achieve 75.66% RMSE reduction compared to RMSE_c, but RMSE_c itself has 5.46% uncertainty. This translates to around 7.22% uncertainty for the RMSE-reduction results (Table 5.1).

Can we reduce the uncertainty by using alternative evaluation metrics like the RMSE, or the percentage of explained variance (`ExplVar`), instead of introducing a control task? In addition to the adjustment for dataset artifacts, the control task provides a baseline to understand the utility. While the RMSE is always positive and `ExplVar` is almost always above 90%, RMSE reduction itself provides a clearer picture of the utility of probing features.

	3 features	7 features	12 features
RTE	5.71%	9.00%	15.16%
COLA	5.46%	10.00%	13.60%
MRPC	5.21%	9.47%	15.63%
SST2	5.03%	9.41%	14.70%
QNLI	5.37%	10.01%	14.07%
QQP	5.80%	9.29%	14.97%

Table 5.5: The relative uncertainties ($\frac{\text{Std}(\text{MSE}_c)}{\text{Mean}(\text{MSE}_c)}$) using 3, 7, and 12 features to regress the 6 fine-tuning task performances.

5.5.9 Can the probing results distinguish the originating language models?

Since the 25 models come from only 5 language models (RoBERTa, XLM, ALBERT, DeBERTa, XLNet), one may wonder if the "model augmentation" procedure "shuffles" the language models sufficiently — if yes, then it would be hard to distinguish the originating language models.

We use 5-class logistic regression from scikit-learn. For any combination of three features, we compute the accuracy following 5-fold cross validation. On all combinations of 3 features, the probing features can reach 0.0027 accuracy (sd=0.0109) better than the random features. This is statistically significant.[12] However, the maximum reachable accuracy is 0.08, whereas even a trivial predictor always outputting "RoBERTa" has an expected 0.24 accuracy (there are 6 RoBERTa models out of 25). The small accuracies show that our "model augmentation" procedure (§5.4.3) produces sufficiently distinct models.

5.6 Additional experimental details

On the effects of random seeds Literature has shown that the choice of random seeds could affect the fine-tuning results a lot (Dodge et al., 2020), and we do not attempt to model this effect. Instead, by fixing the random seed *before* any results are observed, we effectively control for the effect of random seeds in fine-tuning. An alternative experiment setup involves running multiple random seeds — in this case, we will need a mixed-effect model instead of a simple regression model to factor out the effect of random seeds.

Computation budget for fine-tuning The computation budget for fine-tuning varies for the tasks (primarily due to the sizes of the datasets). The time usages for different language models do not differ much — around 16 minutes for RTE, 17 minutes for MRPC, 45 minutes for COLA, 6.5 hours for SST2, 7 hours for QNLI, 18 hours for QQP. If we normalize by the sizes of the datasets, fine-tuning takes between 2 and 6

[12] One-sample one-sided t-test on dof $= 571,703$.

minutes of GPU time (on an RTX 6000 card; we refer to it as "GPU" henceforth) per 1,000 data samples.

Computation budget for probing Before probing, we cache the representations of SentEval data (taking around 60 hours GPU time) to avoid the feed-forward pass, which is the most time-consuming part. Note that we cached the whole SentEval data and then subsample 1,200 per class (around 1%). Caching only the subsampled 1% would take 40 minutes on GPU.

The probing classifications take around 16 hours of CPU time (on an M1 chip; we refer to it as "CPU" henceforth) for each of the 25 models. This includes 7 (probing tasks) $\times$ 12 (layers) $\times$ 7 (probing methods) = 588 (probing configurations).

Consider a configuration of one probing method, 12 layers, and 7 probing tasks. Acquiring 84 probing features would take 40 minutes (caching) + 80 minutes (probing), which is about two hours, where only 1/3 of the two hours need GPU.

Computation in analysis For §5.5.5, the feature selection procedure takes between 350 and 450 seconds for one fine-tuning task when running in RStudio on a laptop (with i7 CPU; we refer to it as RStudio henceforth). All 6 fine-tuning tasks take around 1 hour. Note that this procedure takes around 1/3 time on the M1 CPU.

For §5.5.6, the computation cost is 7 times that of §5.5.2 to §5.5.5, totaling around 7 hours.

For §5.5.7, the probing time is around 15 hours. Subsequent analysis time on RStudio equals §5.5.2 to §5.5.5, i.e., around 1 hour.

For §5.5.8, a Monte Carlo simulation for the uncertainty analysis takes around 15 minutes for all 6 fine-tuning tasks on CPU.

For §5.5.9, running through the features to find the top 3 for distinguishing the language models take 50 minutes on CPU.

RMSE values Table 5.6 shows the RMSE values complementing the RMSE reduction values in Table 5.1. All values appear small in magnitude, but note that comparable scales of RMSE values can be achieved by the random features as well.

5.7 Discussion

Can probing results generalize to non-classification tasks? All fine-tuning tasks and probing tasks in this chapter are text-based classification problems. In the interpretable NLP literature, the probing analyses

	RTE	COLA	MRPC	SST2	QNLI	QQP
All layers one task (§5.5.2)						
BShift	.0353	.0091	.0160	.0050	.0040	.0227
CoordInv	.0336	.0054	.0187	.0044	.0024	.0218
ObjNum	.0427	.0069	.0174	.0036	.0034	.0141
SOMO	.0274	.0074	.0173	.0054	.0040	.0195
Tense	.0430	.0092	.0176	.0055	.0038	.0100
SubjNum	.0489	.0035	.0176	.0044	.0026	.0180
TreeDepth	.0378	.0073	.0207	.0039	.0031	.0204
One layer per task (§5.5.4)	.0380	.0068	.0201	.0048	.0029	.0383
Only three features (§5.5.5)	.0331	.0053	.0149	.0028	.0019	.0125

Table 5.6: RMSE values complementing the RMSE reduction values in Table 5.1.

can also apply to other categories of deep neural networks including translation (Belinkov et al., 2017; Zhang and Bowman, 2018). This generalization across different neural network is intuitive since probing examines the linguistic knowledge encoded in the representations. If a neural model encodes both rich syntactic information (as illustrated by high probing scores in Tense, SubjNum, etc.), and semantic information (as illustrated by high probing scores in BShift, SOMO, etc.) then we will not be surprised when observing that this neural model achieves a high BLEU score. That said, the extent to which probing results remain predictive for BLEU score needs further analysis, which we leave for future works.

Probing is computational-friendly Compared to fine-tuning, probing evaluations require less computation. Fine-tuning the 6 GLUE tasks takes around 30 GPU hours in total, while probing the 7 tasks (all 12 layers) takes 0.7 GPU hours to cache and 1.3 CPU hours to probe. We elaborate the computational budgets in Appendix 5.6. Probing is far more efficient because it does not need to change the parameters in the neural model, and we only need one pass through the neural model and cache the representations. Fine-tuning needs the gradients to update the parameters in the neural models. There are some methods to reduce the computation costs,[13] and we note that probing is competitive as well, in terms of computational time.

Fine-tuning tasks need more specifications Currently, the most popular leaderboards for natural language understanding constitute fine-tuning tasks. A criticism towards these leaderboards is underspecification (D'Amour et al., 2020b) — the short descriptions of the tasks can hardly be inclusive enough to specify the precise abilities required to complete the tasks. To further understand the underspecification problem,

[13]One approach involves using momentum to accelerate the convergence (Kingma and Ba, 2015; Dozat, 2016) Alternatively, the memory usage can be reduced (Gomez et al., 2017; Behrmann et al., 2019). Empirically, limiting the precisions can also accelerate optimization (Shin et al., 2021). Specially-designed structures including Adapters (Houlsby et al., 2019) and LoRA (Hu et al., 2022) are effective as well. Prefix tuning and prompt tuning are lightweight alternatives to fine-tuning (He et al., 2022; Le Scao and Rush, 2021; Li and Liang, 2021). Liu et al. (2021c) summarizes many approaches related to prompt.

researchers recently developed probing datasets (McCoy et al., 2019; Warstadt et al., 2020). Probing results on these datasets have been (indirectly) used in developing deep neural models — performance prediction is a more direct application.

Fine-grained evaluations improve transparency Leaderboard tasks should be customized to the users (Ethayarajh and Jurafsky, 2020). The diversity of probing datasets offers flexible choices to NLP researchers, supporting the diversified considerations to the consumers. Some recently proposed fine-grained leaderboards allow researchers to answer questions like "where does model A outperform model B" (Ma et al., 2021; Narayan et al., 2021; Ruder et al., 2021; Liu et al., 2021b). The probing literature can provide many more datasets for building diverse leaderboards.

Incorporating probing in model developments While the developers are already busy, probing can still bring in benefits to the big model developments, mostly through a multi-dimensional feedback mechanism. The probing datasets introduce targeted knowledge that complement the training datasets of big NLP models. During developments, the model developers can select good model checkpoints to resume or proceed with the help of the probing evaluation scores.

Limitations The evaluation of large language models using only perplexity is uni-dimensional. Evaluations using fine-tuning tasks requires modifying many parameters, hence more costly than probing. Our paper aims at paving the path towards multidimensional evaluations of model parameters within computational budget, so instead of providing a fixed recipe (including fixing the probing datasets and specifying which layers to probe), we provide a general framework and use experiments to show the informativeness and potential utility of the probing results. While probing results are shown to be informative in our experiments, many other methods (e.g., LoRA and prefix-tuning) could optimize similar numbers of parameters. The empirical verifications of other methods are left to future work. Similarly, the problem settings considered in this chapter are all classification problems. We believe this should generalize to other problem settings (e.g., BLEU score on sequential tasks), yet the empirical verifications are left to future work.

5.8 Conclusion

This chapter shows that analyzing probing results can be relevant to developing deep NLP models via predicting a proxy signal, i.e., fine-tuning performance. We show that as few as three probing accuracy scores

can be useful to predict fine-tuning results with RMSEs 40% - 80% smaller than baselines. This can dramatically improve the efficiency of deep learning pipelines. Given several ablation studies, we recommend MLP-20 and RandomForest-100 over other probing methods and show that the probing results from as few as 400 per class may still contain predictability. Probing analysis contain rich resources, and we show their results are closely related to fine-tuning performances. We call for further applications of probing into the developments of deep NLP models.

Chapter 6

Utility of probing from an out-of-domain generalization view

This chapter is based on Zhu et al. (2022a), published at ICML SCIS workshop 2022.

6.1 Introduction

This chapter expands the exploration of the utility of probing. Here I expand the test subject to DNN models beyond language models, and the modality to beyond texts. I consider a setting that is important for developing and deployment of DNNs: out-of-domain generalization.

The training and deployment of deep neural network systems do not always occur in the same environment. During training, the DNNs can perform well on the "source domain", the distribution where the training data come from. We want the DNNs to perform well after we deploy them when the data usually follow a different distribution, i.e., the "target domain". We want the DNNs to maintain their high performance in the target domain. This is the problem of out-of-domain (OOD) generalization, an essential goal in developing DNN systems.

Most existing approaches to developing OOD-generalizable systems follow the invariance principle (Arjovsky et al., 2020), stating that the data representation should allow the optimal predictor performance to match across environments. Two avenues of work stem from this invariance principle. The first avenue focuses on learning a data representation that remains invariant across environments. This goal can translate to regularization terms that minimize the discrepancy across environments (Li et al., 2018a), or auxiliary learning objectives encouraging the representations to be indistinguishable (Ganin et al., 2015; Li et al.,

2018b). The second avenue focuses on letting the optimal predictor performance match across environments. To align the predictors, we can align the gradients (Shahtalebi et al., 2021; Shi et al., 2021; Koyama and Yamaguchi, 2021; Parascandolo et al., 2020) or prevent the predictors to become overly confident (Pezeshki et al., 2021; Wald et al., 2021). This avenue requires a diverse collection of environments, which can be implemented by perturbing the domains by group (Sagawa* et al., 2020), data samples (Krueger et al., 2021; Yan et al., 2020), features (Huang et al., 2020), or some combinations thereof (Huang et al., 2022).

Each paper proposes some improvements against multiple previous algorithms and verifies by evaluating accuracy-based scores in novel domains, including the worst group and leave-one-domain-out accuracy (Gulrajani and Lopez-Paz, 2020; Ye et al., 2021a; Hendrycks et al., 2021). While these evaluation protocols provide a holistic perspective for the performance of the networks, they do not reveal the intrinsic mechanisms of generalization. Many questions remain unanswered, including:

Q1: Do the neural networks arrive at invariant representations somewhere in the network? If yes, where?

Q2: Do the OOD generalization algorithms encourage the neural networks to learn generalizable representations? If yes, how well do these representations generalize?

We argue that, to answer these questions, we should inspect the intermediate representations of the deep neural networks. To attempt answers, we resort to probing, a tool widely used to analyze DNN models. After the deep neural networks are trained, a diagnostic classifier ("probe") is trained to predict a target from the intermediate representations. A higher probing performance indicates the representation is more relevant to the target (Alain and Bengio, 2017). Probing analysis reveals many aspects about the intrinsics of deep neural networks (primarily language models), including the encoded semantic knowledge (Pavlick, 2022) and linguistic structures (Rogers et al., 2020; Manning et al., 2020). For example, Tenney et al. (2019a) found that BERT (Devlin et al., 2019), a Transformer-based language model, automatically forms a pipeline to process textual data in a way that resembles traditional NLP pipelines. The probing results, together with some co-occurrence statistics, can predict the extent to which a feature influences the model's predictions (Lovering et al., 2021). Probing has demonstrated strong potential for examining the intrinsic mechanisms of deep neural networks.

To answer Q1 and Q2, we set up a framework, OOD-Probe, that attaches an auxiliary probing module to the deep neural network models. OOD-Probe predicts the domain attribute from the intermediate representations.

We apply OOD-Probe to 22 algorithms in DomainBed (Gulrajani and Lopez-Paz, 2020). Overall, the

neural networks do not arrive at truly invariant representations. In addition, the probing results reveal interesting patterns that persist across algorithms and differ across datasets. On RotatedMNIST (Ghifary et al., 2015), the lower layers show easier-to-decode domain attributes. On VLCS (Fang et al., 2013) and PACS (Li et al., 2017a), the middle layers show the most easily decodable domain attribute. The higher probing performances also correlate well to the OOD generalization performances. In aggregate, probing performance is predictive of domain generalization performance and provides evaluations in finer granularity. We call for further attention to the models' intrinsics and discuss how probing can help develop OOD generalization systems.

6.2 A neural interpretation of OOD generalization

6.2.1 Problem setting

We want to set up a neural network system that learns a mapping $f : x \in \mathcal{X}_e \rightarrow y \in \mathcal{Y}_e$, for $e \in \mathcal{E}$. In the "leave-one-domain-out" setting of OOD generalization problem, the system f is trained on M training environments $\mathcal{E}_{tr} = \{e_1, e_2, ..., e_M\}$ and tested on a novel environment e_{M+1}.

The neural network f learns a representation Φ to represent the random variable X and preserve rich information about Y. Usually, the neural networks contain a *Featurizer* (e.g., ResNet (He et al., 2016) or multilayer CNNs) and a *Classifier* (e.g., Linear), as shown in Figure 6.1. Both types of *Featurizer* contain multiple intermediate representations. For multilayer CNNs, we probe the representation for each layer. For ResNet, since the residual connections within each block accelerates the passage of information, we probe the representation from each block.

6.2.2 Probing module

The key component in our neural explanation framework is a probing module, which consists of one or more probes (please refer to Chapter 2 for the definition of probe). For a representation Φ of a trained neural network, we attach a probe, which is a *post-hoc* classifier that predicts a predefined target: $f_p : \Phi \rightarrow T$, where the choice of T depends on the (e.g., linguistic) aspect of the representations to be examined (Ettinger et al., 2018; Conneau and Kiela, 2018). We set the target T as the environment label E. Without loss of generality, one can designate alternative targets as the probing target.

Explaining OOD with probing accuracy Many other choices of Perf(f_p) have been discussed in Chapter 2. In this chapter, we will use accuracy. This widely used Perf(f_p) score allows for comparison to the

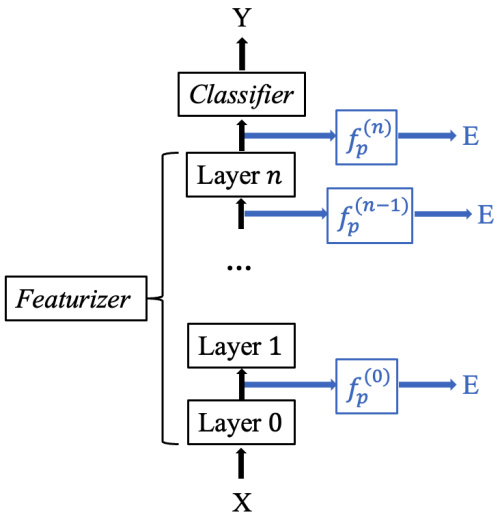


Figure 6.1: A schematic diagram of the proposed framework. The blue parts constitute of the probing module.

performances of OOD generalization algorithms — we elaborate the analyses in Section 6.4.

6.3 Experimental setup

6.3.1 Data

We use four datasets that are widely used in the OOD generalization literature: RotatedMNIST (Ghifary et al., 2015), ColoredMNIST (Arjovsky et al., 2020), VLCS (Fang et al., 2013), and PACS (Li et al., 2017a). These datasets specify classification problems in multiple domains. The variable that denotes the domain, E, affects the joint distributions $P(X, Y)$ in distinct manners.

In RotatedMNIST, E specifies the degree of rotation of the handwritten digits. The E in ColoredMNIST corresponds to the proportion of images assigned a color (red or green). The $f_p : \Phi \rightarrow E$ probing classification performances would be low if the model relies on the digits' shapes but not other factors (e.g., the rotated angle and the colors).

In PACS, E specifies four distinct styles of images — photo, art painting, cartoon, and sketch — all containing images belonging to seven classes: dog, elephant, giraffe, guitar, horse, house, and person. E in VLCS specifies the originating dataset, all of which contain images belonging to five common categories: bird, cat, chair, dog, and person. In both PACS and VLCS, the $f_p : \Phi \rightarrow E$ probing classification performance would be low if the representation describes the content but not the styles of the objects.

6.3.2 OOD Algorithms

We attach the probing module to evaluate 22 DomainBed algorithms using their default hyperparameters. The average test-domain out-split accuracies in leave-one-domain-out evaluation are reported in Table 6.1.

6.3.3 Setting up the probing module

Here we use the most popular type of classifier in the probing literature, a fully connected linear classifier. Compared to more complicated classifiers, linear classifiers are higher in *selectivity* (Hewitt and Liang, 2019), i.e., there is a more significant accuracy difference between highly informative representations and less informative ones (specified by control experiments).

Many algorithms in OOD generalization either use multilayer CNNs or ResNet as the *Featurizer*. The immediate input to the *Classifier* is a D-dimensional vector. All other hidden representations at layer i follow the shape $(C^{(i)}, H^{(i)}, W^{(i)})$, where $C^{(i)}$, $H^{(i)}$, and $W^{(i)}$ are the number of channels, height, and width at the i^{th} layer, respectively. We flatten the representations to $C^{(i)} \times H^{(i)} \times W^{(i)}$-dimensional vectors and input them to the probes.

Two types of *Featurizer* are used in our experiments: 4-layer CNN for RotatedMNIST and ColoredMNIST, and a ResNet-18 for VLCS and PACS. For the former, we attach a probe at the output of each Conv layer. For the latter, we attach a probe at the output of each block. For both types of networks, we add a classifier to probe the output of the *Featurizer*.

6.3.4 Probing procedure

We checkpoint the neural networks trained by OOD generalization algorithms and train the probing classifiers from the frozen checkpoints using the same train/validation splits as the OOD training. The probing also follows a "leave-one-domain-out" setting: to probe a model f trained on domains $\mathcal{E}_{tr}$ (i.e., leaving out e_{M+1}), we train the probing classifier on the same domains to predict $1..M$.

How many data samples are enough for the probes? We used an off-the-shelf script to recommend probing dataset sizes based on the finite function space bound (Zhu et al., 2022c). For the probes we set up, we need between 230k to 270k data samples to bound the uncertainty of the probing accuracy within $\pm 2\%$, and one epoch (containing $N = 5,000$ batches of 64 data samples, totaling 320k data samples) suffices. As an empirical note, most probes' validation accuracies saturate with much fewer than one epoch of samples. Figure 6.4 and 6.5 (in the Appendix) provide some examples.

6.4 Experiment Results

6.4.1 DG algorithms do not remove environment-specific information

Figure 6.3 shows the probing accuracy results Perf(f_p), averaged across all domains in the "leave-one-domain-out" setting. To interpret the accuracy results, let us first define a "dummy classifier" as one that randomly selects a label with uniform probability. On any dataset with n_class distinct labels, the dummy classifier is expected to achieve $\frac{1}{\text{n\_class}}$ accuracy — a dummy accuracy. While a worse accuracy is achievable, it is not a meaningful baseline. We set $\frac{1}{\text{n\_class}}$ to the lower bounds for the colour bars in Figure 6.3.

An accuracy higher than the dummy accuracy by a large margin requires much information from either the data samples or the data distributions. This is observed in all four datasets and most probing tests, indicating that OOD algorithms do not completely remove the environment-specific information. Why is this the case? Here are two alternative hypotheses:

- While the complete removal of environment-specific information is a desirable goal, the domain-invariant representations are hard to arrive at for current DNN-based learning systems.
- In addition to being invariant among environments, the representations should also be highly beneficial for the original classification problem $f : \Phi \to Y$. The multiple optimization goals define a complex game where the optimal representations themselves follow a "trade-off" among these goals.

6.4.2 Layerwise patterns across algorithms

Following the rows of Figure 6.3, consistent patterns are observable. Perf(f_p) decreases as we probe the upper layers for RotatedMNIST. The probing results on ColoredMNIST show a similar pattern, but this trend is not as consistent across algorithms. Note that the probing performances are only slightly above the dummy accuracy at all layers.

On VLCS and PACS, an "increase-then-decrease" pattern is visible as we probe into higher layers. The representations from the middle blocks of ResNet-18 encode the domain information in easier-to-decode manners than the remaining blocks. Note that the representation from a block, $\Phi^{(i+1)}$, is *processed* from the representation of the previous block, $\Phi^{(i)}$. Therefore, the data processing inequality guarantees that the information about the environment does not decrease as the block number i increases, i.e., $I(\Phi^{(i)}\,;\,E) \geq I(\Phi^{(i+1)}\,;\,E)$. In other words, the ResNets do not increase the domain information — they encode the domain information more *linearly* in the middle parts.

Note that the algorithms with similar generalization performance Perf(f_g) could have markedly differ-

ent probing performances Perf(f_p). For example, CDANN and CORAL achieve Perf(f_g) of 0.97 and 0.98, respectively, on RotatedMNIST. Their probing performances on lower layers are similar, but on the last hidden representation, they get Perf$(f_p)^{(4)}$ of 0.55 and 0.26, respectively. When the models do not generalize, their representations might encode more or less linearly readable information about the environments.

6.4.3 Layerwise correlations to the OOD performance

To further understand the utility of the probing performances, we compute the Pearson correlation[1] of Perf(f_p) and Perf(f_g). Note that the algorithms with values markedly ($\geq 3\sigma$ where σ is the std in Table 6.1) different from the results report by the DomainBed paper are removed.[2] Table 6.2 (in Appendix) shows the results. Additionally, Figure 6.2 in the Appendix shows the correlation heatmaps. The lower layers of networks on RotatedMNIST show strong correlations, indicating that a linear encoding of the rotation might be beneficial for the generalization. The linear encoding of the styles on PACS shows a similar correlation trend, as reflected by the high correlation values on probes 2 and 3. This trend is slightly different for VLCS, where the Probe_2 results, which show the highest Perf(f_p), do not correlate as strongly to the OOD performance.

The above correlations might be affected by some confounding factors, e.g., the inclusion of individual algorithms. To control for this factor, we also compute the correlation w.r.t each algorithm. Tables 6.3-6.6 (in Appendix) show the results. In general, Perf(f_p) are positively correlated to Perf(f_g) for RotatedMNIST, ColoredMNIST, and PACS. The correlations are mostly negative for VLCS.

6.5 Discussion

Setting up probing targets The environment attribute E is the most convenient target to use, but whether it is the most suitable probing target to set up remains an open question. Our intuition is that a feature T can be a better alternative for the environment attribute E if it can specify the data shift between the environments *concretely*. Otherwise, one can always use the environment attribute. The interpretable NLP literature has many attempts to run carefully controlled trials that use features to specify the difference between the environments (Warstadt and Bowman, 2020; McCoy et al., 2019; Kaushik et al., 2019). While this approach to constructing datasets is expensive, it allows the designated features to serve as informative probing targets.

[1] using `scipy.stats.pearsonr`

[2] These include: IB_IRM for RotatedMNIST, CDANN, DANN, IB_IRM, IRM for PACS.

Fine-grained evaluations for generalization The findings from OOD-Probe open up several possibilities for improving OOD generalization using fine-grained evaluation signals, at least in the following two ways. First, since different modules in the network demonstrate different mechanisms for processing domain-related information, targeted designs may be beneficial. For example, objective functions can be set up to encourage learning linear encoding of target features in lower layers. Second, probing allows a convenient feedback "dashboard" for viewing the effects of design choices in building DNN models.

Privacy, fairness, and societal impacts The problem of learning and evaluating high-quality representations that are invariant across domains has broad societal impacts. In privacy, the "domain attribute" can be the personal identity. This problem can translate to "removing the speaker attribute from voice while keeping the sounds recognizable" (Tomashenko et al., 2022). If the "domain attribute" refers to the demographic property, the problem can be formulated as learning a rich and fair representation (Zemel et al., 2013). The algorithms that remove protected attributes can have profound long-term impacts on multiple groups (Liu et al., 2018; Khani and Liang, 2021) — we defer to Mehrabi et al. (2022) for a summary.

6.6 Conclusion

While the probing analysis has revealed a wide collection of findings on language models, their utility on the non-LM DNNs deserves more attention. This chapter expands on the range of probing analysis by studying the OOD generalization problem. We propose OOD-Probe, a flexible framework that inspects the neural networks and provides layerwise scores regarding their encoding of the domain attributes. We find patterns that differ across several OOD datasets but remain relatively stable across many algorithms on DomainBed. The probing results show correlation and predictability to the generalization performance, opening up future paths to integrate the probing of DNNs and the development of generalizable DNNs.

6.7 Tables and Figures

For readability, the tables and plots showing experiment results are grouped together in this section.

Algorithm	RotatedMNIST	ColoredMNIST	VLCS	PACS
ANDMask (Parascandolo et al., 2020)	0.96 ± 0.03	0.55 ± 0.30	0.69 ± 0.18	0.79 ± 0.10
CAD (Ruan et al., 2021)	0.97 ± 0.02	0.51 ± 0.33	0.67 ± 0.12	0.71 ± 0.12
CDANN (Li et al., 2018b)	0.97 ± 0.03	0.54 ± 0.30	0.53 ± 0.27	0.18 ± 0.08
CORAL (Sun and Saenko, 2016)	0.98 ± 0.01	0.51 ± 0.23	0.73 ± 0.16	0.83 ± 0.07
CondCAD (Ruan et al., 2021)	0.97 ± 0.02	0.49 ± 0.33	0.66 ± 0.08	0.77 ± 0.09
DANN (Ganin et al., 2015)	0.97 ± 0.03	0.53 ± 0.33	0.44 ± 0.26	0.21 ± 0.05
ERM (Vapnik, 1991)	0.98 ± 0.02	0.52 ± 0.25	0.74 ± 0.16	0.83 ± 0.07
GroupDRO (Sagawa* et al., 2020)	0.98 ± 0.02	0.55 ± 0.22	0.73 ± 0.15	0.83 ± 0.07
IB_ERM (Ahuja et al., 2021)	0.98 ± 0.01	0.50 ± 0.29	0.74 ± 0.14	0.79 ± 0.08
IB_IRM (Ahuja et al., 2021)	0.20 ± 0.07	0.51 ± 0.01	0.49 ± 0.10	0.13 ± 0.07
IRM (Arjovsky et al., 2020)	0.70 ± 0.12	0.56 ± 0.06	0.49 ± 0.10	0.16 ± 0.09
MLDG (Li et al., 2017b)	0.97 ± 0.02	0.52 ± 0.27	0.73 ± 0.15	0.86 ± 0.07
MMD (Li et al., 2018a)	0.98 ± 0.02	0.37 ± 0.23	0.72 ± 0.18	0.82 ± 0.06
MTL (Blanchard et al., 2021)	0.98 ± 0.01	0.53 ± 0.26	0.74 ± 0.14	0.85 ± 0.06
Mixup (Yan et al., 2020)	0.97 ± 0.02	0.52 ± 0.35	0.76 ± 0.16	0.83 ± 0.08
RSC (Huang et al., 2020)	0.97 ± 0.03	0.53 ± 0.33	0.74 ± 0.10	0.80 ± 0.10
SD (Pezeshki et al., 2021)	0.98 ± 0.02	0.51 ± 0.25	0.74 ± 0.17	0.81 ± 0.09
SagNet (Nam et al., 2021)	0.98 ± 0.02	0.52 ± 0.29	0.75 ± 0.15	0.83 ± 0.07
SelfReg (Kim et al., 2021)	0.98 ± 0.01	0.51 ± 0.31	0.76 ± 0.14	0.82 ± 0.08
TRM (Xu and Jaakkola, 2021)	0.98 ± 0.02	0.53 ± 0.32	0.65 ± 0.02	0.83 ± 0.06
VREx (Krueger et al., 2021)	0.98 ± 0.02	0.56 ± 0.29	0.74 ± 0.15	0.86 ± 0.06

Table 6.1: Domain generalization leave-one-domain-out accuracy ($\pm$ std)

Data	Probe 0	Probe 1	Probe 2	Probe 3	Probe 4	Probe 5
RotatedMNIST	0.7955**	0.8982**	0.7451**	0.4026	0.0268	N/A
ColoredMNIST	-0.1859	-0.0489	0.4389*	0.3646	0.2607	N/A
VLCS	0.6781**	-0.1489	0.1147	0.8437**	0.8434**	0.8732**
PACS	-0.5936**	0.6171**	0.7291**	0.9703**	0.8202**	0.6592**

Table 6.2: Pearson correlations of probing performance Perf(f_p) and domain generalization performance Perf(f_g). * and ** indicate $p < 0.05$ and $p < 0.01$, respectively.

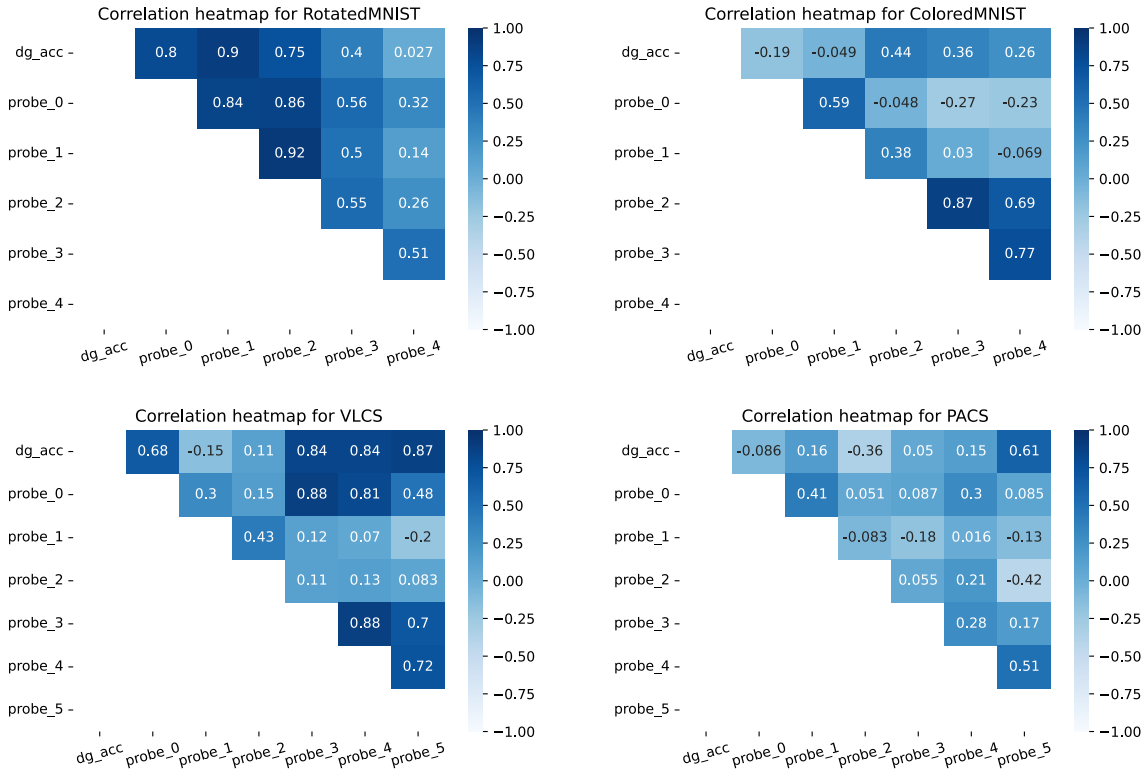


Figure 6.2: Correlation within the probing results, and those between the probing performances and the OOD generalization performances.

Algorithm	Probe_0	Probe_1	Probe_2	Probe_3	Probe_4	Probe_5
ANDMask	0.2881	0.9437***	0.9572***	0.9483***	0.9964***	N/A
CAD	0.6126	0.9242***	0.856**	0.8381**	0.8411**	N/A
CDANN	0.8551**	0.83**	0.9461***	0.9482***	0.9499***	N/A
CORAL	0.8307**	0.8858**	0.9213***	0.9198***	0.0953	N/A
CondCAD	0.9311***	0.9198***	0.9481***	0.8669**	0.5268	N/A
DANN	0.6012	0.8587**	0.9222***	0.921***	0.9667***	N/A
ERM	0.8494**	0.8367**	0.8013*	0.8496**	0.9335***	N/A
GroupDRO	0.7032	0.8758**	0.8952**	0.8978**	0.9108**	N/A
IB_ERM	0.6786	0.6929	0.8788**	0.7546*	0.3893	N/A
IB_IRM	0.4827	0.6954	0.724	0.4284	0.0707	N/A
IRM	0.4979	0.8738**	0.8717**	0.9316***	0.8896**	N/A
MLDG	0.7268	0.9349***	0.9366***	0.9632***	0.8557**	N/A
MMD	0.7999*	0.8775**	0.933***	0.9493***	0.6374	N/A
MTL	0.8572**	0.8059*	0.8892**	0.8879**	0.9716***	N/A
Mixup	0.5907	0.8608**	0.6677	0.7965*	0.9684***	N/A
RSC	0.7001	0.9388***	0.9051**	0.9155**	0.9747***	N/A
SD	0.917**	0.913**	0.9409***	0.9573***	0.7983*	N/A
SagNet	0.7862*	0.8598**	0.9141**	0.9094**	0.9466***	N/A
SelfReg	0.9142**	0.8514**	0.9222***	0.9134**	0.7575*	N/A
TRM	0.6009	0.8282**	0.8852**	0.9188***	0.9054**	N/A
VREx	0.6077	0.8968**	0.8604**	0.8569**	0.9847***	N/A

Table 6.3: Correlations between probing results and generalization accuracies, on RotatedMNIST

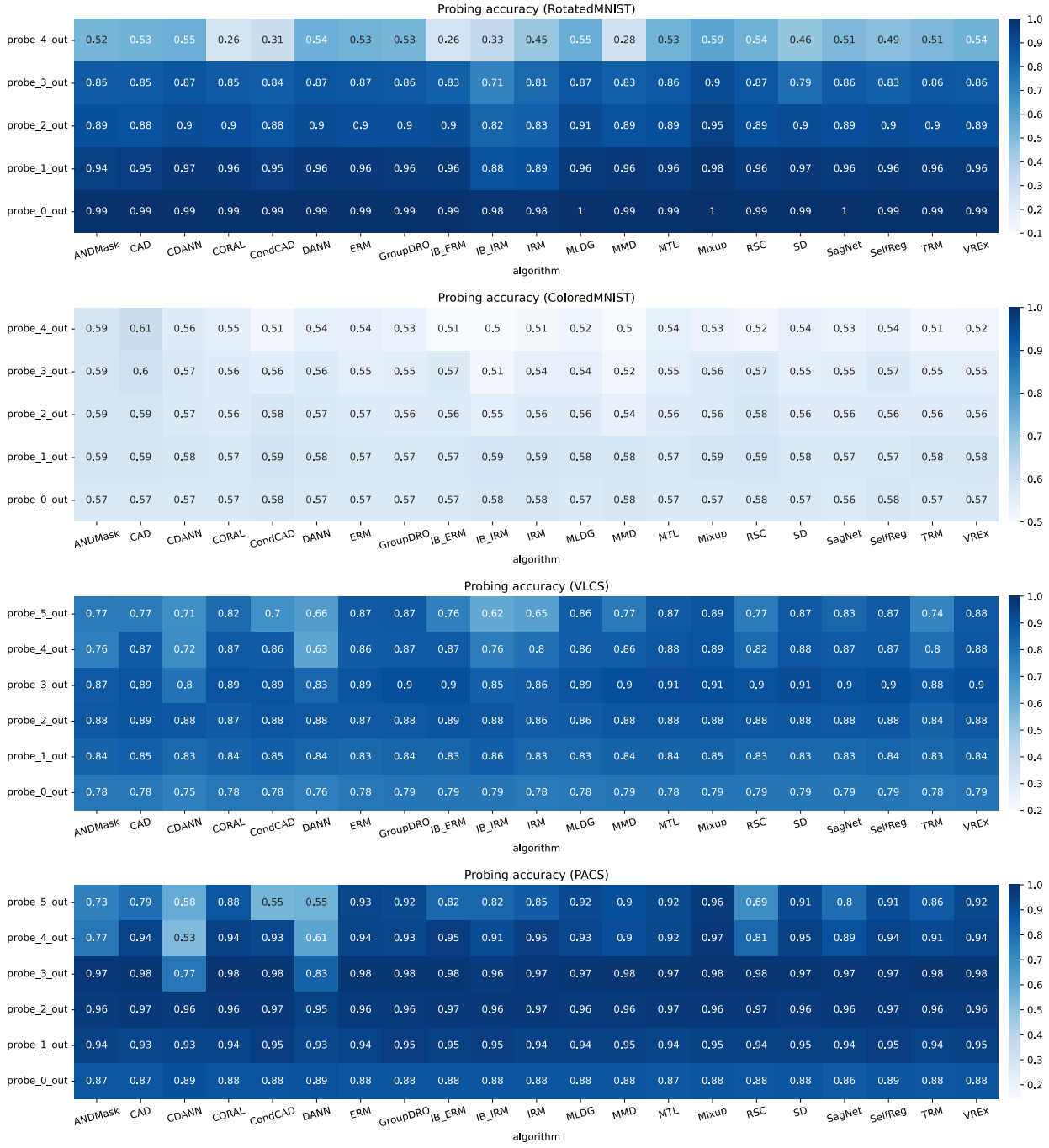


Figure 6.3: Probing accuracies on four datasets. The color bars scale from $\frac{1}{\text{n\_class}}$ to 1.00.

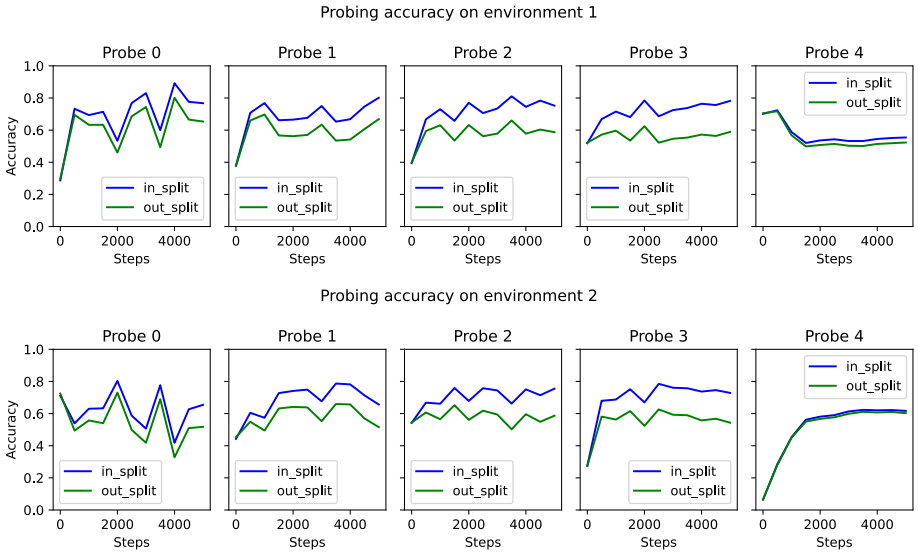


Figure 6.4: Probing accuracy on a checkpoint of ERM on ColoredMNIST. The test environment of ERM is 0, which is left out by the probe.

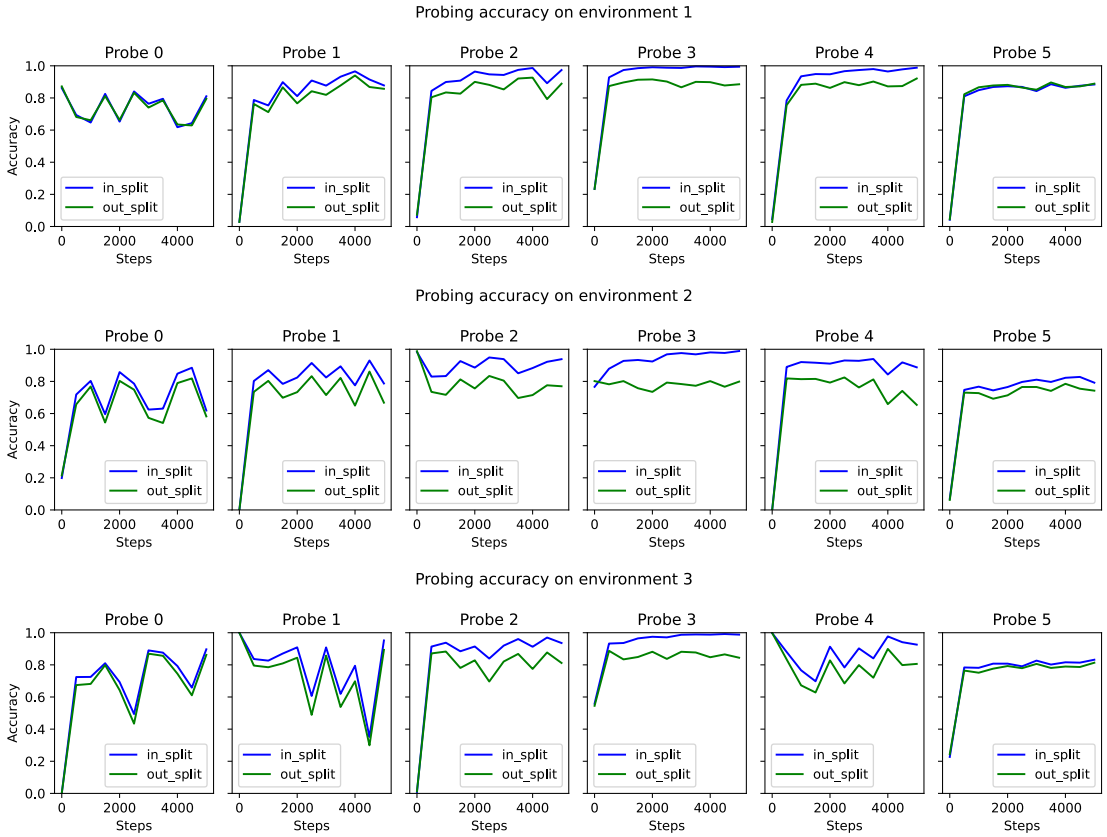
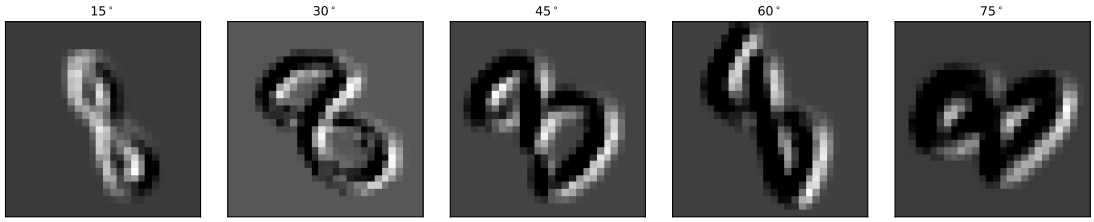
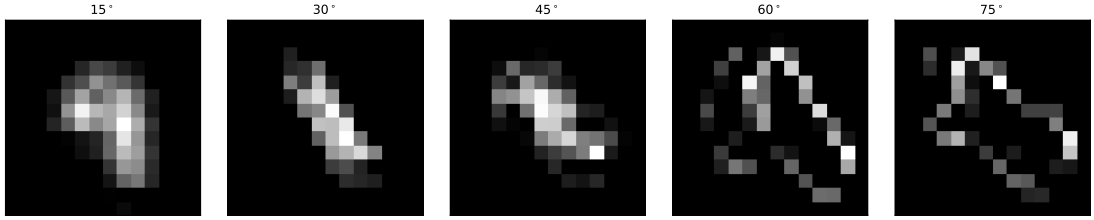


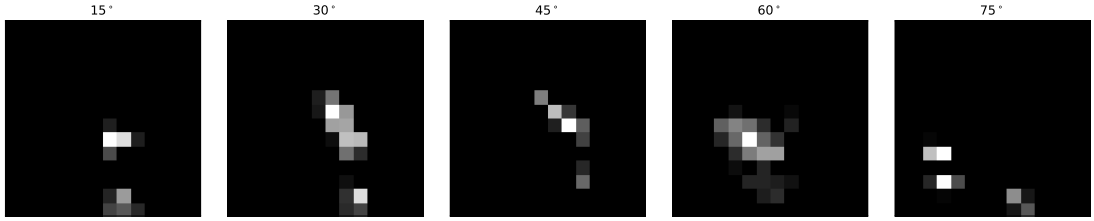
Figure 6.5: Probing accuracy on a checkpoint of ERM on VLCS with test environment set to 0.



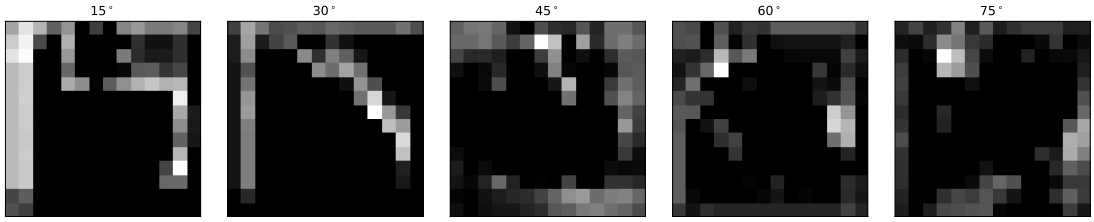
(a) Probe 0 (28×28). It is relatively easy to tell apart both the originating domain (rotated angles) and the contents.



(b) Probe 1 (14×14). The contents are less obvious, but the degrees of rotations are still visible.



(c) Probe 2 (14×14). The representations appear more abstract than the previous layers, but we can still guess the rotation angles to some extent.



(d) Probe 3 (14×14). Neither the digit contents nor the rotation angles remain intelligible. However, the probing classifiers can reach .75 accuracy.

Figure 6.6: Intermediate representations of Rotated MNIST (trained by ERM) as the inputs of the first four probes. The fifth probe takes one-dimensional inputs, whereas only two-dimensional representations are intelligible via plotting. Each representation contains 64, 128, 128, and 128 channels, respectively. Only one channel is (randomly) sampled to visualize.

Algorithm	Probe_0	Probe_1	Probe_2	Probe_3	Probe_4	Probe_5
ANDMask	0.9697	0.986	0.9852	0.9888*	0.9981**	N/A
CAD	0.9747	0.9725	0.9585	0.9872	0.9896*	N/A
CDANN	0.9585	0.9726	0.9725	0.9981**	0.9623	N/A
CORAL	0.9903*	0.9992**	0.9968*	0.9993**	0.9908*	N/A
CondCAD	0.9827	0.9997**	0.9928*	0.6784	0.2138	N/A
DANN	0.9263	0.9764	0.9781	0.9818	0.7047	N/A
ERM	0.9762	0.9873	0.9933*	0.9874	0.9907*	N/A
GroupDRO	0.9788	0.9885*	0.9858	0.9776	0.9818	N/A
IB_ERM	0.9879*	0.9761	0.9955*	0.9847	0.9747	N/A
IB_IRM	0.2722	0.2686	0.1684	0.1637	0.7736	N/A
IRM	0.945	0.9233	0.983	0.9998**	0.9792	N/A
MLDG	0.9782	0.9993**	0.9953*	0.9908*	0.9861	N/A
MMD	0.9185	0.995*	0.9824	0.7451	1.0***	N/A
MTL	0.9722	0.9907*	0.9868	0.9881*	0.999**	N/A
Mixup	0.971	0.9871	0.9861	0.9983**	0.7062	N/A
RSC	0.9578	0.9568	0.979	0.9941*	-0.0878	N/A
SD	0.9787	0.9948*	0.9837	1.0***	0.9861	N/A
SagNet	0.982	0.9935*	0.9822	0.9837	0.9255	N/A
SelfReg	0.9868	0.9951*	0.9892*	0.9777	0.7666	N/A
TRM	0.991*	0.9978**	0.9942*	0.9377	0.2663	N/A
VREx	0.9688	0.9866	0.9787	0.9851	0.5169	N/A

Table 6.4: Correlations between probing results and generalization accuracies, on ColoredMNIST

Algorithm	Probe_0	Probe_1	Probe_2	Probe_3	Probe_4	Probe_5
ANDMask	-0.7884	-0.6042	-0.8627	-0.7988	-0.6038	-0.9108*
CAD	-0.8111	-0.6155	-0.6242	-0.7604	-0.7486	-0.9077*
CDANN	0.0623	-0.5973	-0.3463	0.5724	0.021	0.5191
CORAL	-0.6874	-0.3425	-0.4622	-0.6697	-0.7194	-0.6265
CondCAD	-0.9467*	-0.8162	-0.863	-0.967**	-0.9185*	-0.9941***
DANN	-0.1123	-0.629	-0.5488	-0.6396	-0.3297	0.4269
ERM	-0.8021	-0.4361	-0.7221	-0.8069	-0.8837	-0.8699
GroupDRO	-0.8521	-0.7036	-0.7736	-0.8923	-0.9307*	-0.9052*
IB_ERM	-0.8316	-0.5454	-0.7996	-0.8347	-0.8475	-0.8376
IB_IRM	-0.9672**	-0.9477*	-0.9136*	-0.9915***	-0.9974***	-0.7454
IRM	-0.9979**	-0.8173	-0.943	-0.9651	-0.956	0.445
MLDG	-0.7728	-0.6106	-0.6124	-0.8167	-0.8326	-0.821
MMD	-0.7461	-0.6395	-0.5214	-0.7459	-0.8597	-0.716
MTL	-0.8281	-0.8388	-0.7541	-0.711	-0.6164	-0.7218
Mixup	-0.6426	-0.5704	-0.5358	-0.6617	-0.7821	-0.7691
RSC	-0.7308	-0.4961	-0.5818	-0.8237	-0.9211*	-0.9132*
SD	-0.6835	-0.3245	-0.6887	-0.7215	-0.7497	-0.7583
SagNet	-0.6156	-0.8272	-0.7361	-0.7811	-0.8167	-0.9139*
SelfReg	-0.85	-0.7936	-0.7676	-0.7544	-0.6486	-0.7837
TRM	0.5303	0.8719	0.9031*	0.8972	0.8852	0.9105*
VREx	-0.7578	-0.6201	-0.7837	-0.7981	-0.762	-0.8093

Table 6.5: Correlations between probing results and generalization accuracies, on VLCS

Algorithm	Probe_0	Probe_1	Probe_2	Probe_3	Probe_4	Probe_5
ANDMask	0.7009	0.6549	0.7471	0.7242	0.7655	0.9147*
CAD	0.7815	0.9024*	0.7055	0.854	0.6974	0.3663
CDANN	-0.9258*	-0.9026*	-0.7547	0.8968	0.4159	0.7199
CORAL	0.6937	0.6337	0.8343	0.7439	0.9382*	0.7638
CondCAD	0.661	0.7221	0.7567	0.7183	0.7373	0.9398*
DANN	0.1647	0.2556	0.0702	-0.202	0.2035	0.0368
ERM	0.4843	0.5695	0.6677	0.6186	0.4791	0.6495
GroupDRO	0.5355	0.6534	0.6629	0.641	0.6643	0.665
IB_ERM	0.741	0.8657	0.8592	0.7541	0.7057	0.6374
IB_IRM	0.7409	0.6835	0.71	0.8454	0.8313	0.8211
IRM	0.8565	0.9172*	0.9833**	0.9654**	0.9758**	0.9398*
MLDG	0.8648	0.9347*	0.9774**	0.8492	0.9193*	0.9298*
MMD	0.5435	0.6747	0.6089	0.529	0.6636	0.6518
MTL	0.5355	0.7426	0.6701	0.6356	0.6907	0.7749
Mixup	0.7521	0.7421	0.8298	0.6567	0.8161	0.8569
RSC	0.5571	0.6241	0.7671	0.6141	0.6833	0.0018
SD	0.4503	0.4382	0.7056	0.5861	0.5235	0.5208
SagNet	0.2259	0.2972	0.3456	0.1986	-0.8632	-0.7221
SelfReg	0.4667	0.6937	0.772	0.7706	0.6388	0.8693
TRM	0.3149	0.6426	0.5857	0.6102	0.5043	0.3968
VREx	-0.1246	0.0428	0.0869	-0.0261	-0.08	0.1654

Table 6.6: Correlations between probing results and generalization accuracies, on PACS

Chapter 7

Conclusion

Probing, the use of (usually small) machine learning models to study deep neural network models, has led to intriguing findings. In this dissertation, I described how the validity, reliability, and utility of probing can be improved via systematic analysis.

Chapter 2 described the scope of probing analysis for the thesis, and briefly reviewed some methods for probing DNNs.

Chapter 3 considered the validity problem from an information-theoretic view. I derived the terms stating the difference between the measured values and the desired values (the mutual information), for two control settings. While the error terms for the control probing settings are still nonzero, these are better than the settings with the "one-classifier" probing setting. This leads to my recommendation that a control mechanism is beneficial for the validity of probing.

Chapter 4 considered the reliability problem. With the generalization bounds in machine learning theory, I derive a data requirement of probing from an intuitive reliability requirement: that the comparison result between two probing tests remains consistent with high probability. This data size recommendation is empirically supported by statistical power analysis.

Chapters 5 and 6 proceed to improve the utility of probing. I show that the findings of the probing tests can be used to predict the downstream performance of the models. Additionally, the probing test results are correlated to the generalization performance of the deep neural network models.

I consider the following future directions particularly worth exploring.

Strengthening theoretical foundations for probing In addition to the validity and reliability, many more aspects of probing can be solidified theoretically. Recently, there have been some arguments about the

limitations of current probing techniques. Kumar et al. (2022) mentioned that the probing classifiers tend to rely on spurious patterns. Another recent paper, Geiger et al. (2021), cast doubt on whether the probing classifiers themselves are capable of capturing the causal dependencies between concepts. The efficacy of amnesic probing (Elazar et al., 2021) indicates that these limitations may not be fundamental, but rather some limitations in the existing task specifications of probing. However, more thorough theoretical and empirical studies are necessary in the future.

Incorporating probing to model developments Probes can be integrated as a part of the development procedures of DNN models. Nowadays, the model developers are already quite busy, and the marginal utility of probing to model developers could be diminishing. In response to reviewers in Zhu et al. (2022b), we discussed a possibility: to use the probing results to assess the qualities of the checkpoints and to decide which checkpoint to resume training from subsequently.

Are there additional utilities for the model users? One way that probing can be useful is to be part of a *self-explaining* model, where the model's decisions are accompanied by the probe's outputs (which could provide contexts and rationales for the model's decisions). Considering that the probes can be optimized using a fraction of the computation powers of the model itself, this approach appears to be an efficient one to make the models self-explaining (as opposed to, e.g., re-training the model with some self-explaining components from scratch). Another efficient approach is to prompt the model to verbally explain its decisions ("chain-of-thought") (Wei et al., 2022). This approach has some limitations, as the generated explanations may not usually match the true decision procedures of the models (Ye and Durrett, 2022). Yet there are some recent approaches that improve faithfulness (Lyu et al., 2023; Lanham et al., 2023).

Probing and model control Intuitively, after probing analysis, which identifies the part of the model that is responsible for a task, we should be equipped with insights into how to change the model's behavior. Recent works like REMEDI (Hernandez et al., 2023) adapt the targets of probes to learn transformations to the representations. Differentiable masking (Cao et al., 2021; De Cao et al., 2021) hints at another approach where probing can be applied to model control. Probing leverages backpropagation, which allows us to automatically find a subset of model parameters that are crucial for the tasks. The subset could be as small as a few neurons (Dai et al., 2022), and the intervention on this subset can be achieved with reasonable computation resources. I believe that the control of model behavior can lead to safer and more trustworthy AI systems.

Granularities of interpretability A recent avenue related to the works presented in this thesis is mechanistic interpretability. Compared to probing, mechanistic interpretability offers several additional granularities of analysis. Probing operates on the representation level, and mechanistic interpretability involves neuron-level, module-level, and circuit-level analysis.

On the neuron level, Elhage et al. (2022) discusses the potential configuraitons that each neuron is a "superposition" of multiple functional primitives. A later work, Bricken et al. (2023), attempted to decompose these functional primitives using sparse autoencoders. Dai et al. (2022) proposed the "knowledge neuron" thesis, and Niu et al. (2023) further studied the roles of the knowledge neurons. On the module level, Geva et al. (2021) showed that the key-value pairs in the Transformer networks have important knowledge storage roles. Dar et al. (2023) found that the embedding space of the Transformers are related to the stored vocabulary. On the circuit level, Elhage et al. (2021) proposed a math framework to study the Transformer networks. More recently, Conmy et al. (2023) attempted to automatically discover a circuit from a neural network. Merullo et al. (2023) found that many neural network circuits of different tasks share some common components. Going forward, I believe a combination of neuron-level, circuit-level, and representation-level analysis can bring richer understanding towards the mechanisms of neural networks.

Bibliography

Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. 2021. Invariance Principle Meets Information Bottleneck for Out-of-Distribution Generalization. *arXiv 2106.06607 [cs,stat]*.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual String Embeddings for Sequence Labeling. In *COLING*, pages 1638–1649.

Guillaume Alain and Yoshua Bengio. 2017. Understanding intermediate layers using linear classifier probes. In *ICLR*.

Vamsi Aribandi, Yi Tay, and Donald Metzler. 2021. How reliable are model diagnostics? In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1778–1785, Online. Association for Computational Linguistics.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2020. Invariant Risk Minimization. *arXiv:1907.02893 [cs, stat]*.

Stéphane Aroca-Ouellette, Cory Paik, Alessandro Roncone, and Katharina Kann. 2021. PROST: Physical reasoning about objects through space and time. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4597–4608, Online. Association for Computational Linguistics.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. 2019. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR.

Frank B Baker and Seock-Ho Kim. 2004. *Item response theory: Parameter estimation techniques*. CRC Press.

Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. 2019. Invertible Residual Networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 573–582. PMLR.

Yonatan Belinkov. 2021. Probing Classifiers: Promises, Shortcomings, and Alternatives. *arXiv:2102.12452*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Yonatan Belinkov, Sebastian Gehrmann, and Ellie Pavlick. 2020. Interpretability and analysis in neural NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 1–5, Online. Association for Computational Linguistics.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Gilles Blanchard, Aniket Anand Deshmukh, Urun Dogan, Gyemin Lee, and Clayton Scott. 2021. Domain Generalization by Marginal Transfer Learning. *arXiv:1711.07910 [stat]*.

Samuel R. Bowman. 2023. Eight things to know about large language models.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2023/monosemantic-features/index.html.

David A Broniatowski. 2021. Psychological foundations of explainability and interpretability in artificial intelligence. Technical Report NIST IR 8367, National Institute of Standards and Technology (U.S.), Gaithersburg, MD.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss,

Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Steven Cao, Victor Sanh, and Alexander Rush. 2021. Low-Complexity Probing via Finding Subnetworks. In *NAACL-HLT*, pages 960–966, Online. Association for Computational Linguistics.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020a. With Little Power Comes Great Responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Dallas Card, Peter Henderson, Urvashi Khandelwal, Robin Jia, Kyle Mahowald, and Dan Jurafsky. 2020b. With little power comes great responsibility. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9263–9274, Online. Association for Computational Linguistics.

Mingda Chen, Zewei Chu, and Kevin Gimpel. 2019. Evaluation benchmarks and learning criteria for discourse-aware sentence representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 649–662, Hong Kong, China. Association for Computational Linguistics.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards Automated Circuit Discovery for Mechanistic Interpretability.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An Evaluation Toolkit for Universal Sentence Representations. In *LREC*. ArXiv: 1803.05449.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of ACL*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Lee J Cronbach. 1951. Coefficient alpha and the internal structure of tests. *psychometrika*, 16(3):297–334.

Paweł Czyż, Frederic Grabowski, Julia E. Vogt, Niko Beerenwinkel, and Alexander Marx. 2023. Beyond Normal: On the Evaluation of Mutual Information Estimators.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, and others. 2020a. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. 2020b. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2023. Analyzing transformers in embedding space.

Nicola De Cao, Leon Schmid, Dieuwke Hupkes, and Ivan Titov. 2021. Sparse Interventions in Language Models with Differentiable Masking.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2185–2194, Hong Kong, China. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. Technical Report arXiv:2002.06305, arXiv.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Timothy Dozat. 2016. Incorporating Nesterov momentum into Adam. *OpenReview*.

Ellen A Drost. 2011. Validity and reliability in social science research. *Education Research and Perspectives*, 38(1):105–123.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic Probing: Behavioral Explanation with Amnesic Counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/toy$_model/index.html$.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4846–4853, Online. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. 2018. Assessing Composition in Sentence Vector Representations. *arXiv:1809.03992 [cs]*.

Allyson Ettinger, Ahmed Elgohary, and Philip Resnik. 2016. Probing for semantic evidence of composition by means of simple classification tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 134–139, Berlin, Germany. Association for Computational Linguistics.

Chen Fang, Ye Xu, and Daniel N. Rockmore. 2013. Unbiased Metric Learning: On the Utilization of Multiple Datasets and Web Images for Softening Bias. In *2013 IEEE International Conference on Computer Vision*, pages 1657–1664, Sydney, Australia. IEEE.

Amir Feder, Katherine A. Keith, Emaad Manzoor, Reid Pryzant, Dhanya Sridhar, Zach Wood-Doughty, Jacob Eisenstein, Justin Grimmer, Roi Reichart, Margaret E. Roberts, Brandon M. Stewart, Victor Veitch, and Diyi Yang. 2022. Causal Inference in Natural Language Processing: Estimation, Prediction, Interpretation and Beyond. *Transactions of the Association for Computational Linguistics*, 10:1138–1158.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. 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 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2015. Domain-Adversarial Training of Neural Networks. *JMLR*.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. 2021. Causal Abstractions of Neural Networks.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. 2015. Domain Generalization for Object Recognition with Multi-task Autoencoders. *arXiv:1508.07680 [cs, stat]*.

Nahid Golafshani. 2003. Understanding reliability and validity in qualitative research. *The qualitative report*, 8(4):597–607.

Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. 2017. The reversible residual network: Backpropagation without storing activations. *Advances in neural information processing systems*, 30.

Ishaan Gulrajani and David Lopez-Paz. 2020. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*.

Abhijeet Gupta, Gemma Boleda, Marco Baroni, and Sebastian Padó. 2015. Distributional vectors encode referential attributes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 12–21, Lisbon, Portugal. Association for Computational Linguistics.

Rowan Hall Maudslay, Josef Valvoda, Tiago Pimentel, Adina Williams, and Ryan Cotterell. 2020. A Tale of a Probe and a Parser. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7389–7395, Online. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a Unified View of Parameter-Efficient Transfer Learning. In *International Conference on Learning Representations*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In *International Conference on Learning Representations*.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. 2021. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8320–8329, Montreal, QC, Canada. IEEE.

Evan Hernandez, Belinda Z. Li, and Jacob Andreas. 2023. Measuring and Manipulating Knowledge Representations in Language Models.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher Manning. 2021. Conditional probing: measuring usable information beyond a baseline. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1626–1639, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and Interpreting Probes with Control Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Zeyi Huang, Haohan Wang, Dong Huang, Yong Jae Lee, and Eric P. Xing. 2022. The Two Dimensions of Worst-case Training and the Integrated Effect for Out-of-domain Generalization. In *arXiv:2204.04384 [cs]*.

Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. 2020. Self-Challenging Improves Cross-Domain Generalization. In *ECCV*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2177–2190, Online. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. In *ICLR*.

Fereshte Khani and Percy Liang. 2021. Removing spurious features can hurt accuracy and affect groups disproportionately. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 196–205.

Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. 2023. Causal Reasoning and Large Language Models: Opening a New Frontier for Causality.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Daehee Kim, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. 2021. SelfReg: Self-supervised Contrastive Regularization for Domain Generalization. *arXiv:2104.09841 [cs]*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*. ArXiv:1412.6980.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. *Advances in neural information processing systems*, 28.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3849–3864, Online. Association for Computational Linguistics.

Masanori Koyama and Shoichiro Yamaguchi. 2021. When is invariance useful in an Out-of-Distribution Generalization problem? *arXiv:2008.01883*.

Helena Chmura Kraemer. 1992. Measurement of reliability for categorical data in medical research. *Statistical Methods in Medical Research*, 1(2):183–199. PMID: 1341657.

Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-Distribution Generalization via Risk Extrapolation (REx). In *Proceedings of the 38th International Conference on Machine Learning*, pages 5815–5826. PMLR.

Artur Kulmizev, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre. 2020. Do neural language models show preferences for syntactic formalisms? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4077–4091, Online. Association for Computational Linguistics.

Abhinav Kumar, Chenhao Tan, and Amit Sharma. 2022. Probing Classifiers are Unreliable for Concept Removal and Detection. *arxiv:2207.04153*, page 40.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. Measuring faithfulness in chain-of-thought reasoning.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*, 521(7553):436–444.

Quentin Lhoest, Albert Villanova del Moral, Patrick von Platen, Thomas Wolf, Yacine Jernite, Abhishek Thakur, Lewis Tunstall, Suraj Patil, Mariama Drame, Julien Chaumond, Julien Plu, Joe Davison, Simon

Brandeis, Teven Le Scao, Victor Sanh, Kevin Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Steven Liu, Nathan Raw, Sylvain Lesage, Théo Matussière, Lysandre Debut, Stas Bekman, and Clément Delangue. 2021. Huggingface/datasets: 1.12.1.

Bai Li, Zining Zhu, Guillaume Thomas, Frank Rudzicz, and Yang Xu. 2022. Neural reality of argument structure constructions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7410–7423, Dublin, Ireland. Association for Computational Linguistics.

Bai Li, Zining Zhu, Guillaume Thomas, Yang Xu, and Frank Rudzicz. 2021. How is BERT surprised? Layerwise detection of linguistic anomalies. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4215–4228, Online. Association for Computational Linguistics.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2017a. Deeper, Broader and Artier Domain Generalization. *arXiv:1710.03077 [cs]*.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2017b. Learning to Generalize: Meta-Learning for Domain Generalization. *arXiv:1710.03463 [cs]*.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. 2018a. Domain Generalization with Adversarial Feature Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5400–5409, Salt Lake City, UT. IEEE.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Ya Li, Mingming Gong, Xinmei Tian, Tongliang Liu, and Dacheng Tao. 2018b. Domain Generalization via Conditional Invariant Representation. *arXiv:1807.08479 [cs, stat]*.

Percy Liang. 2016. CS229T/STAT231: Statistical Learning Theory (Winter 2016).

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In *Proceedings of the 2020*

Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6862–6868, Online. Association for Computational Linguistics.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019. Open Sesame: Getting Inside BERT's Linguistic Knowledge. *ACL BlackBoxNLP Workshop*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *TACL*, 4:521–535.

Leo Z. Liu, Yizhong Wang, Jungo Kasai, Hannaneh Hajishirzi, and Noah A. Smith. 2021a. Probing Across Time: What Does RoBERTa Know and When? In *Findings of EMNLP*, pages 820–842, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lydia T. Liu, Sarah Dean, Esther Rolf, Max Simchowitz, and Moritz Hardt. 2018. Delayed Impact of Fair Machine Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3150–3158. PMLR.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic Knowledge and Transferability of Contextual Representations. 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Pengfei Liu, Jinlan Fu, Yang Xiao, Weizhe Yuan, Shuaichen Chang, Junqi Dai, Yixin Liu, Zihuiwen Ye, and Graham Neubig. 2021b. ExplainaBoard: An explainable leaderboard for NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 280–289, Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021c. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692*. ArXiv: 1907.11692.

Charles Lovering, Rohan Jha, Tal Linzen, and Ellie Pavlick. 2021. Predicting Inductive Biases of Pre-Trained Models. In *ICLR*.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful Chain-of-Thought Reasoning.

Zhiyi Ma, Kawin Ethayarajh, Tristan Thrush, Somya Jain, Ledell Wu, Robin Jia, Christopher Potts, Adina Williams, and Douwe Kiela. 2021. Dynaboard: An Evaluation-As-A-Service Platform for Holistic Next-Generation Benchmarking. In *Conference on Neural Information Processing Systems*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Christopher D. Manning, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2022. A Survey on Bias and Fairness in Machine Learning. Technical Report arXiv:1908.09635, arXiv.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2023. Circuit Component Reuse Across Tasks in Transformer Language Models.

Alessio Miaschi, Dominique Brunato, Felice Dell'Orletta, and Giulia Venturi. 2020. Linguistic Profiling of a Neural Language Model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 745–756, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. 2017. Variational Dropout Sparsifies Deep Neural Networks. In *Proceedings of International Conference of Machine Learning*, page 10.

Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. 2021. Reducing Domain Gap by Reducing Style Bias. *arXiv:1910.11645 [cs]*.

Avanika Narayan, Piero Molino, Karan Goel, Willie Neiswanger, and Christopher Ré. 2021. Personalized Benchmarking with the Ludwig Benchmarking Toolkit. In *Conference on Neural Information Processing Systems*.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2023. What does the Knowledge Neuron Thesis Have to do with Knowledge? In *ICLR*.

Jingcheng Niu, Wenjie Lu, Eric Corlett, and Gerald Penn. 2022. Using Roark-Hollingshead distance to probe BERT's syntactic competence. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xing Niu, Prashant Mathur, Georgiana Dinu, and Yaser Al-Onaizan. 2020. Evaluating robustness to input perturbations for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8538–8544, Online. Association for Computational Linguistics.

Jekaterina Novikova. 2021. Robustness and Sensitivity of BERT Models Predicting Alzheimer's Disease from Text. *W-NUT @ EMNLP 2021*.

Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele, and Bernhard Schölkopf. 2020. Learning explanations that are hard to vary. *arXiv:2009.00329 [cs, stat]*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Ellie Pavlick. 2022. Semantic Structure in Deep Learning. *Annual Review of Linguistics*, 8. Publisher: Annual Reviews.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, and David Cournapeau. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, page 6.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. 2021. Gradient Starvation: A Learning Proclivity in Neural Networks. *arXiv:2011.09468 [cs, math, stat]*.

Tiago Pimentel and Ryan Cotterell. 2021. A Bayesian framework for information-theoretic probing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2869–2887, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020a. Pareto Probing: Trading Off Accuracy for Complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153, Online. Association for Computational Linguistics.

Tiago Pimentel, Naomi Saphra, Adina Williams, and Ryan Cotterell. 2020b. Pareto probing: Trading off accuracy for complexity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3138–3153, Online. Association for Computational Linguistics.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020c. Information-Theoretic Probing for Linguistic Structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4609–4622, Online. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. BERT is Not a Knowledge Base (Yet): Factual Knowledge vs. Name-Based Reasoning in Unsupervised QA. *arXiv:1911.03681*.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null It Out: Guarding Protected Attributes by Iterative Nullspace Projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256, Online. Association for Computational Linguistics.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. 2021. Probing the Probing Paradigm: Does Probing Accuracy Entail Task Relevance? *EACL*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics.

Brian Roark and Kristy Hollingshead. 2008. Classifying chart cells for quadratic complexity context-free inference. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 745–752, Manchester, UK. Coling 2008 Organizing Committee.

Pedro Rodriguez, Joe Barrow, Alexander Miserlis Hoyle, John P. Lalor, Robin Jia, and Jordan Boyd-Graber. 2021. Evaluation examples are not equally informative: How should that change NLP leaderboards? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4486–4503, Online. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Yangjun Ruan, Yann Dubois, and Chris J. Maddison. 2021. Optimal Representations for Covariate Shift. *ICLR*.

Sebastian Ruder, Noah Constant, Jan Botha, Aditya Siddhant, Orhan Firat, Jinlan Fu, Pengfei Liu, Junjie Hu, Dan Garrette, Graham Neubig, and Melvin Johnson. 2021. XTREME-R: Towards More Challenging and Nuanced Multilingual Evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10215–10245, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally Robust Neural Networks. In *International Conference on Learning Representations*.

Soroosh Shahtalebi, Jean-Christophe Gagnon-Audet, Touraj Laleh, Mojtaba Faramarzi, Kartik Ahuja, and Irina Rish. 2021. SAND-mask: An Enhanced Gradient Masking Strategy for the Discovery of Invariances in Domain Generalization. *arXiv:2106.02266 [cs]*.

Claude Elwood Shannon. 1948. A Mathematical Theory of Communication. *The Bell system technical journal*, 27(3):379–423.

Yuge Shi, Jeffrey Seely, Philip H. S. Torr, N. Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. 2021. Gradient Matching for Domain Generalization. *arXiv:2104.09937*.

Dongyeob Shin, Geonho Kim, Joongho Jo, and Jongsun Park. 2021. Low complexity gradient computation techniques to accelerate deep neural network training. *IEEE Transactions on Neural Networks and Learning Systems*.

Aviv Slobodkin, Leshem Choshen, and Omri Abend. 2021. Mediators in determining what processing BERT performs first. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 86–93, Online. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Anirudh Srinivasan, Sunayana Sitaram, Tanuja Ganu, Sandipan Dandapat, Kalika Bali, and Monojit Choudhury. 2021. Predicting the Performance of Multilingual NLP Models.

Baochen Sun and Kate Saenko. 2016. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. *arXiv:1607.01719 [cs]*.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, et al. 2021. ERNIE 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation. *arXiv preprint arXiv:2107.02137*.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? Probing for sentence structure in contextualized word representations. In *ICLR*.

Natalia Tomashenko, Xin Wang, Emmanuel Vincent, Jose Patino, Brij Mohan Lal Srivastava, Paul-Gauthier Noé, Andreas Nautsch, Nicholas Evans, Junichi Yamagishi, Benjamin O'Brien, Anaïs Chanclu, Jean-François Bonastre, Massimiliano Todisco, and Mohamed Maouche. 2022. The VoicePrivacy 2020 Challenge: Results and findings. *Computer Speech & Language*, 74.

Lucas Torroba Hennigen, Adina Williams, and Ryan Cotterell. 2020. Intrinsic probing through dimension selection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 197–216, Online. Association for Computational Linguistics.

A. M. Turing. 1950. Computing Machinery and Intelligence. *Mind*, LIX(236):433–460.

Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1141–1158, Online. Association for Computational Linguistics.

Vladimir Vapnik. 1991. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4.

Elena Voita and Ivan Titov. 2020. Information-Theoretic Probing with Minimum Description Length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, Online. Association for Computational Linguistics.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7222–7240, Online. Association for Computational Linguistics.

Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. 2021. On Calibration and Out-of-Domain Generalization. In *NeurIPS*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *International Conference on Learning Representations*.

Zirui Wang, Adams Wei Yu, Orhan Firat, and Yuan Cao. 2021. Towards zero-label language learning. *arXiv preprint arXiv:2109.09193*.

Alex Warstadt and Samuel R. Bowman. 2020. Can neural networks acquire a structural bias from raw linguistic data? Technical Report arXiv:2007.06761, arXiv.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. arXiv:2201.11903.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. 2020. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8625–8646, Online. Association for Computational Linguistics.

Xiaoyu Xing, Zhijing Jin, Di Jin, Bingning Wang, Qi Zhang, and Xuanjing Huang. 2020. Tasty burgers, soggy fries: Probing aspect robustness in aspect-based sentiment analysis. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3594–3605, Online. Association for Computational Linguistics.

Yilun Xu and Tommi Jaakkola. 2021. Learning Representations that Support Robust Transfer of Predictors. *arXiv:2110.09940 [cs]*.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. *ICLR*.

Yadollah Yaghoobzadeh, Katharina Kann, T J Hazen, Eneko Agirre, and Hinrich Schütze. 2019. Probing for Semantic Classes: Diagnosing the Meaning Content of Word Embeddings. In *ACL*, pages 5740–5753, Florence, Italy. Association for Computational Linguistics.

Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. 2020. Improve Unsupervised Domain Adaptation with Mixup Training. *arXiv:2001.00677 [cs, stat]*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Gregory Yauney and David Mimno. 2021. Comparing text representations: A theory-driven approach. *arXiv preprint arXiv:2109.07458*.

Nanyang Ye, Kaican Li, Lanqing Hong, Haoyue Bai, Yiting Chen, Fengwei Zhou, and Zhenguo Li. 2021a. OoD-Bench: Benchmarking and understanding out-of-distribution generalization datasets and algorithms. *arXiv:2106.03721*.

Xi Ye and Greg Durrett. 2022. The Unreliability of Explanations in Few-shot Prompting for Textual Reasoning. In *NeurIPS*.

Zihuiwen Ye, Pengfei Liu, Jinlan Fu, and Graham Neubig. 2021b. Towards more fine-grained and reliable NLP performance prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3703–3714, Online. Association for Computational Linguistics.

Daniel Zeman, Joakim Nivre, and Mitchell et al Abrams. 2019. Universal dependencies 2.5. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361, Brussels, Belgium. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting Few-sample BERT Fine-tuning. In *International Conference on Learning Representations*.

Yichu Zhou and Vivek Srikumar. 2021. DirectProbe: Studying representations without classifiers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5070–5083, Online. Association for Computational Linguistics.

Zining Zhu, Aparna Balagopalan, Marzyeh Ghassemi, and Frank Rudzicz. 2021. Quantifying the Task-Specific Information in Text-Based Classifications. *arXiv preprint arXiv:2110.08931*.

Zining Zhu, Haoming Jiang, Jingfeng Yang, Sreyashi Nag, Chao Zhang, Huang Jie, Yifan Gao, Frank Rudzicz, and Bing Yin. 2023. Situated Natural Languages Explanations. In *ACL NLRSE Workshop*.

Zining Zhu, Chuer Pan, Mohamed Abdalla, and Frank Rudzicz. 2020. Examining the rhetorical capacities of neural language models. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 16–32, Online. Association for Computational Linguistics.

Zining Zhu and Frank Rudzicz. 2020. An information theoretic view on selecting linguistic probes. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9251–9262, Online. Association for Computational Linguistics.

Zining Zhu, Soroosh Shahtalebi, and Frank Rudzicz. 2022a. OOD-Probe: A Neural Interpretation of Out-of-Domain Generalization. *ICML SCIS Workshop*.

Zining Zhu, Soroosh Shahtalebi, and Frank Rudzicz. 2022b. Predicting fine-tuning performance with probing. In *EMNLP*. Association for Computational Linguistics.

Zining Zhu, Jixuan Wang, Bai Li, and Frank Rudzicz. 2022c. On the data requirements of probing. In *Findings of the Association of Computational Linguistics*. Association for Computational Linguistics.