METAPROBE: A Representation- and Task-Agnostic Probe

Yichu Zhou
School of Computing
University of Utah
flyaway@cs.utah.edu

Vivek Srikumar
School of Computing
University of Utah
svivek@cs.utah.edu

Abstract

Probing contextualized representations typically involves comparing task-specific model predictions against ground truth linguistic labels. Although this methodology shows *what* information can be recovered by a classifier, it does not reveal *how* a classifier uses the representation to make its decision. To address the latter problem, we ask: Do task-classifiers rely on representation- and task-independent geometric patterns in the embedding space? We explore this question by developing METAPROBE, an approach that uses geometric properties of representations to predict the behavior of task-specific classifiers (i.e, their predictions as opposed to the ground truth). Our experiments reveal the existence of universal geometric patterns across representations that can predict classifier predictions. Consequently, this allows us to posit a geometric explanation for the impressive performance of contextualized representations.

1 Introduction

Pre-trained contextualized representations (e.g., Peters et al., 2018; Devlin et al., 2019; Liu et al., 2019b; He et al., 2021) have advanced the state-of-the-art across NLP. Unsurprisingly, probing representations, i.e., understanding what linguistic information they encode, and how they do so, has provoked much recent interest in the NLP research community (e.g., Hupkes and Zuidema, 2018; Tenney et al., 2019; Voita and Titov, 2020; Lasri et al., 2022; Immer et al., 2022; Belinkov, 2022; Choudhary et al., 2022; Wang et al., 2023).

Although previous work undoubtedly helps understand contextualized representations, they suffer from two major drawbacks. First, they do not offer a justification of *how* classifiers trained on a representation arrive at their decisions. For example, the commonly used paradigm of classifier-based probing treats representations as black boxes, and relies on the final predictive performance to infer the knowledge encoded in them. Second, since previous work probes representations individually, they do not uncover common patterns in classifiers' decisions across tasks and representations. The fact that most contextualized representations work well on across tasks (Liu et al., 2019a) leads us to conjecture the existence of universal patterns in the decision processes of classifiers that can be uncovered by examining representations.

In this work, we ask: *Are there universal geometric patterns in how contextualized representations embed text that explain their impressive performance across tasks?* To explore this question, we propose a new supervised probing paradigm that studies *predictions* of multiple task-specific classifiers (henceforth *task-classifiers*) based on geometric properties of representations. By mimicking the behavior of task-classifiers in this fashion, we discover and analyze the patterns in the representations that lead to task-classifier decisions.

We present METAPROBE, an instantiation of this idea of exploring **universal geometric** patterns in contextualized representations. Given an unseen example and a representation, METAPROBE estimates how probable it is for a trained task-classifier to predict each label. Importantly, it does so only using features that use geometric properties of the representation and the task data. In other words, instead of trying to mimic the consensus of annotators as standard classifiers do, METAPROBE mimics other classifiers trained on the same representation and data. This probability distribution, coupled with the choice of features, allows us to analyze the usefulness of a representation for predicting an example. For example, a uniform label distribution for an example indicates that the current representation is not informative for that example because classifiers will be maximally confused about it. METAPROBE is a linear model; consequently, its parameters help understand which geometric properties are relevant for task-classifiers.

Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP, pages 233–249

Our experiments on ten NLP datasets (covering five different tasks) and four contextualized representations reveal task- and representation-agnostic patterns in how the geometry of embeddings is employed by successful models. We show that these common patterns can serve as "zero-shot probes" for unseen tasks (manifested as datasets) and representations. Finally, by inspecting the learned METAPROBE parameters, we find that the task-classifiers make their decisions using not only simple geometric properties such as distances between examples (which is usually employed by a nearest neighbor probe), but also more sophisticated geometric properties, e.g. involving potential changes to the decision surfaces that separate two different labels (which *cannot* be captured by a nearest neighbor probe).

In summary, the contributions of this work are:

1. We propose a new supervised probing paradigm of predicting the behavior of task-classifiers using the geometric structure of examples in an embedding space. We instantiate this idea with a linear model called METAPROBE.
2. We hypothesize and verify the existence of universal geometric patterns over contextualized representations that can explain the predictions of classifiers trained over them.
3. Via experiments, we show that METAPROBE identifies important geometric properties that account for the predictive decision process of a task-classifier, even on previously unseen tasks and representations.

2 Two Characteristics of Representations

Contextualized representations have two characteristics: their predictiveness and their descriptive intrinsic properties. Existing probing methodologies focus on one or the other (Michael et al., 2020). *Predictive probing* involves training supervised classifiers to predict one or more specific linguistic properties using the given representations. Representation quality is evaluated using various criteria, e.g. accuracy, complexity, etc., derived from the learned classifiers (e.g. Conneau et al., 2018; Kim et al., 2019; Kassner and Schütze, 2020; Goodwin et al., 2020; Pimentel et al., 2020b,a; Aghazadeh et al., 2022; Tucker et al., 2022; Gonen et al., 2022; Arps et al., 2022). In contrast, *descriptive probing* looks into the intrinsic structure of representations (Ethayarajh, 2019; Zhou and Srikumar, 2021; Xypolopoulos et al., 2021; Chang et al., 2022), and focuses on discovering properties of the representation using cluster analysis (Aharoni and Goldberg, 2020) or visualization technqiues (Reimers et al., 2019; Vig, 2019).

2.1 Predictiveness of Representations

A large class of today's NLP applications involve building a classifier (denoted by h henceforth) using a dataset (denoted by D) with a pre-trained contextualized representation such as $\text{RoBERTa}_{\text{large}}$ (denoted by ϕ) as a feature space. As a result, a representation's predictiveness—that is, the performance of a classifier trained over it—has received most attention in the probing literature (e.g., Xiang et al., 2022; Evci et al., 2022, and many others). However, Zhou and Srikumar (2021) point out that using the performance of a single task-classifier may not relibly characterize the quality of a representation because other confounding factors may affect its performance. Instead, the entire set of classifiers H learned over the representation may provide more information. With such a set, we can ask several natural questions about an unseen example x:

- What label will the majority of the classifiers in H predict for x?
- Which labels are confusable for x? That is, which labels will be predicted by at least one of the classifiers in H?
- Suppose we draw a random classifier from H. What is the probability that it will predict a label y?[1]

The first question is commonly investigated in predictive probing work by taking the average accuracy of multiple runs. It directly shows how well a contextualized representation captures a linguistic property. The second question can tell us how confidently a representation encodes an example x—something that the answer to the first question does not reveal. The last question tells us how difficult the example x is. For example, if the probability that a randomly chosen classifier in H predicts a label is uniformly distributed over the label set (see footnote 1), then the representation ϕ is not informative for x *even though* many individual classifiers in H may be correct in their predictions.

[1]Note that the probability described here is different from the distribution over labels predicted by any single classifier. Here, the label uncertainty is due to the choice of classifiers.

2.2 Descriptive Properties of Representations

One shortcoming of predictive probing is that it treats the representation as a blackbox. It cannot reveal how representations help learned classifiers to make their decisions. This question is the focus of another line of work, called *descriptive probing* (e.g., Reif et al., 2019; Voita et al., 2019; Saphra and Lopez, 2019), and we refer to the attributes uncovered by such work as the *descriptive intrinsic properties of a representation*. Descriptive probing analyzes a representation on its own terms by finding patterns in the representations *without* reference to any specific task. We can ask:

- Are there any task-agnostic regularities in a representation?
- Do they correlate with an NLP task?
- If so, how do they contribute to the predictive behavior of the representation?

However, without a target task, descriptive probing fails to connect these structural patterns with the predictiveness of a representation.

2.3 Quantifying Predictiveness and Descriptive Properties

To understand how task-classifiers make their decisions for unseen examples, we propose to use descriptive aspects of a representation to estimate its predictiveness for a target task. We first need to quantify these two characteristics.

Quantifying Predictiveness. As discussed in §2.1, we care about how different task-classifiers behave on the given representation. Beyond the usual source of uncertainty stemming from the data, we also have uncertainty over the choice of classifiers (Hewitt and Liang, 2019; Talmor et al., 2020; Zhou and Srikumar, 2021). One way to address this is to "marginalize away" the uncertainty by enumerating all (or, many of) the possible classifiers. That is, we can quantify the predictiveness of a representation by having many classifiers make predictions on the same example. The empirical distribution over their predictions captures the predictiveness of the representation for this example. We will refer to this label distribution for each unseen example x as *predicted label distribution*:

$$P_{pred}(y \mid x) = \frac{1}{|H|} \sum_{h \in H} \mathbb{1}[h(x) = y] \quad (1)$$

Here, the set H is the set of classifiers learned from a dataset D using a representation ϕ, and the notation $\mathbb{1}[\cdot]$ represents the indicator function.

As an illustration, given an example and 100 task-classifiers, suppose 90 classifiers predict label `A`, 5 predict `B`, and 5 predict `C`. The empirical distribution of interest is the distribution that allocates 90% probability to `A` and 5% each to `B` and `C`.

Two points are worth noting about P_{pred}. First, P_{pred} is a property of a representation ϕ and an example x, and not any specific task-classifier. Second, P_{pred} does not require the ground truth label. It describes how task-classifiers would behave given the representation and an example.

Quantifying Descriptive Intrinsic Properties. To quantify the descriptive properties of a representation, we use geometric attributes based on DIRECTPROBE (Zhou and Srikumar, 2021), a recently proposed supervised clustering technique. Given a representation ϕ and training set D, DIRECTPROBE returns a set of linearly separable clusters of examples, which we will call $\mathcal{C} = \{C_i\}_{i=1}^{n}$. Each cluster C_i is guaranteed to contain examples from D that have *the same label*. There may be multiple clusters for a label as the labels may not be linearly separable in the given representation. Importantly, any decision boundary separating the labels must cross the regions between differently labeled clusters. Abstractly, we can think of the clusters as forming a geometric layout of labels in the embedding space. Subsequent work by Zhou and Srikumar (2022) showed that the margin between two clusters correlates with the how well task-classifiers using the representation generalize. In this work, we take a step further to explore other geometric properties beyond the margin.[2]

3 From Descriptive Properties to Predictiveness

In this section, we first present METAPROBE, a linear model that uses the geometric features of a representation to predict the predicted label distribution (from §2.3) for an example. Then, we introduce the geometric features we will explore in this work. Finally, we describe details about the training process of METAPROBE.

3.1 Modeling Predicted Labels

Given a pair of dataset and representation (D, ϕ), suppose q is the predicted label distribution (eq. (1)) for an unseen example x. Let $\mathcal{C} = \{C_i\}_{i=1}^{n}$ denote the set of DIRECTPROBE clusters for the data. We

[2]We use the DIRECTPROBE implementation from https://github.com/utahnlp/DirectProbe.

define METAPROBE to be a linear model that uses only properties of the clusters $\mathcal{C}$ and an example x to predict the predicted label distribution q.

Before getting into the details of METAPROBE, we first give a high level overview. Recall from §2.3 that each DIRECTPROBE cluster is associated with one label, but a label may be spread across multiple clusters. For an unseen example, a natural prediction strategy is to use the nearest cluster and its label, which was explored by Zhou and Srikumar (2021). Rather than predicting a single label, we seek to construct a distribution over all labels (i.e. the predicted label distribution). To do so, we consider all clusters. For each cluster C_i, we compute a score that indicates the affinity between x and C_i. Since a label may correspond to multiple DIRECTPROBE clusters, we score labels by aggregating the scores of clusters for each label, before normalizing to obtain a valid probability distribution over the labels. To operationalize this idea, we extract geometric features $\psi(x, C_i) \in \mathbb{R}^d$ that characterize the relationship of the example x to the cluster C_i (described in §3.2). The score of x and C_i is the weighted combination of its features:

$$s(x, C_i) = W^\top \psi(x, C_i) \tag{2}$$

where $W \in \mathbb{R}^d$ is a learned parameter vector that is shared by all clusters.

Let $\mathbf{v} \in \mathbb{R}^n$ be the score vector for all n clusters, i.e. $\mathbf{v}_i = s(x, C_i)$. While we may normalize the scores using the softmax function σ to obtain a probability distribution $\sigma(\mathbf{v}) \in \mathbb{R}^n$ over clusters, we are interested in the distribution over labels rather than over clusters. We resolve this mismatch and score each label using the sum of scores for all its clusters. To do so, we introduce a matrix $M \in \{0, 1\}^{n \times m}$ (where m is the number of labels):

$$M_{ij} = \begin{cases} 1 & \text{if } i\text{-th cluster has } j\text{-th label} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

We project the scores over clusters to the scores over labels by multiplying the score vector by M. Then, we can compute the predictive label distribution for x as softmax$(\mathbf{v} \cdot M)$.

3.2 Geometric Features for METAPROBE

In the discussion that follows, we will assume that for an example x, the clusters in $\mathcal{C}$ are indexed their closeness to x. That is, C_1 is closer to x than C_2 and so on.[3] Also, for brevity, we use the notation x to denote both an example and its representation $\phi(x)$; the context will make it clear whether we are referring to an example or a point in the representation space. For each pair (x, C_i), we extract features to decide if x should be merged by cluster C_i. We consider three classes of features: cluster-only features, distance features, and merging features.

Cluster-only Features. We first consider the properties of the cluster C_i by itself. For example, such features allow us to investigate how the number of examples in the cluster affects the decision of a task-classifier. Specifically, we consider four properties for each cluster C_i: (i) The number of examples it contains; (ii) the average pairwise distances between them; (iii) the standard deviation of the pairwise distances; and (iv) the distances between the cluster C_i and other clusters.

Distance Features. Intuitively, the cluster-only features are insufficient to explain the decision of a task-classifier for an example x. Distances between x and C_i are also informative. For example, if x is inside the cluster C_i (having zero distance), any task-classifier will likely assign it the same label as C_i. We have four distance features: (i) An indicator for whether x is inside the convex hull of points in C_i; (ii) the distance between x and the convex hull of C_i; (iii) the distance between x and the centroid of C_i; (iv) the distance between x and the span of a pair of clusters (C_i, C_j), for $1 \leq j \leq k$. [4] We include the distances between x and the span of a pair of clusters because this information might be helpful to find confusing labels for a task-classifier.

Merging Features. Apart from the cluster-only and distance features, we also consider how much geometry will change if we merge x into the cluster C_i. For example, suppose we find that including x in C_i produces a convex hull in the embedding space that overlaps with some other existing cluster belonging to a different label. (Recall that DIRECTPROBE clusters are linearly separable, and hence non-overlapping.) In this case, a task-classifier will probably not predict label of C_i. We use three sets of features to estimate the change caused by merging x to C_i: (i) An indicator for whether x can

[3] In this work, all distances are Euclidean. Other distances may offer novel geometric insights, and studying them is a direction of future work.

[4] This feature follows the intuition that C_j could be a strong competitors for C_i if x lies in the span of (C_i, C_j)

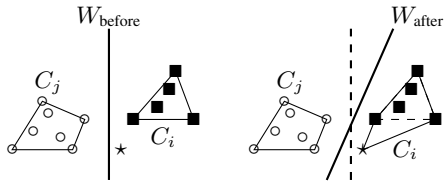

Figure 1: Diagram of the geometry. $\star$ marks the unseen example. C_i and C_j are two clusters in the neighborhood of the unseen example. C_i is the current cluster we are considering. Left (right) subfigure shows the layouts before (after) merging the unseen example to C_i. W_{before} and W_{after} are the max-margin hyperplanes between C_i and C_j before and after merging.

be merged by C_i without breaking the linear separablability condition; (ii) the change in distance between the merged x and C_i and another cluster C_j for $1 \leq j \leq k, i \neq j$; (iii) the change of the max-margin hyperplanes between the merged x and C_i and C_j for $1 \leq j \leq k, i \neq j$.

As an illustration, let us look at the last of the merging features, Figure 1 shows a simple example. It shows the max-margin hyperplanes before and after merging an example $\star$ to C_i, represented by W_{before} and W_{after}, respectively. The last merging feature is defined as $\frac{\|W_{before} - W_{after}\|_1}{\|W_{before}\|_1 + \|W_{after}\|_1}$. Appendix F gives additional details for all features.

3.3 Training METAPROBE

Next, we will discuss the process of training METAPROBE, summarized as Algorithm 1 and Figure 2. After training, we use the learned linear classifier (parameterized by W) to make predictions for unseen examples as described in §3.1.

Task-classifier Set. Since METAPROBE seeks to mimic the predicted label distribution from eq. (1), we need to estimate the distribution for one or more dataset-representation pairs. Recall that we need a set of task-classifiers H to compute the distribution. To this end, we train a large set of classifiers—from simple linear classifiers to two-layer neural networks with various activation functions—on the training split of each dataset and representation, resulting in N task-classifiers in total per setting [5], i.e., $|H| = N$ (Algorithm 1, line 4). Appendix C gives more details about this undertaking.

Sampling for Training Set. Next, we need a collection of examples (points in the embedding space) that can be paired with the empirical predicted label distribution. But we cannot use the task training set for this purpose: not only are the classifiers above trained on it, we also use it to construct DIRECTPROBE clusters for feature extraction. To resolve this issue, we note that we seek to examine the geometry of the representation space, and not the text that led to the embeddings. Moreover, the task classifiers operate on embeddings, and are unaware of whether those embeddings are derived from linguistically meaningful text. Consequently, we can use any points in the embedding space, provided they are distributionally similar to real data points.

Following this observation, we sample $20k$ points in the embedding space by linearly interpolating training examples. Let v_0 and v_1 be two randomly chosen vectors from the training set (embedded by a representation). We sample a random point v_s in the span of these two vectors. That is,

$$v_s = \alpha v_0 + (1 - \alpha) v_1 \tag{4}$$

where α is uniformly sampled from $[0, 1]$. These points need not correspond to a linguistically meaningful textual input, but since they live in the span of training points, our trained classifiers can make predictions on them.

We use the learned task-classifiers to make predictions for the sampled vectors and compute the predicted label distribution using eq. (1). Note that there is no ground truth for these sampled vectors; we seek to understand how task-classifiers make their decisions—a process independent of the ground truth. These sampled $20k$ vectors and their predicted label distributions, form the training data for METAPROBE (Algorithm 1, lines 4-6).

Optimization. Finally, to complete the discussion on training, let us look at the loss function whose minimization will provide the weights W that define METAPROBE. Let q denote the target *predicted label distribution* for an example x, and $\mathbf{v}$ denote its score vector as defined by eq. (2). We use the Kullback-Leibler divergence between the two label distributions as our loss:

$$\ell(q, p) = KL(q, p) = KL(q, \text{softmax}(\mathbf{v} \cdot M)) \tag{5}$$

We use standard optimization tools for training and optimization. Appendix D has details.

4 Representations and Tasks

In this section, we describe the English representations and tasks we will use in our experiments.

[5] In our experiments, $N = 1010$.

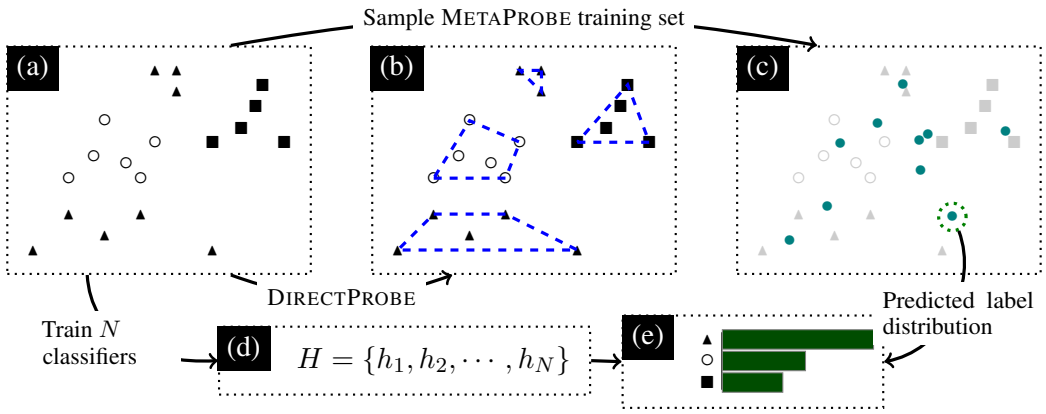

Figure 2: An illustration of training steps for METAPROBE. (a) A dataset with three labels (○, ■, ▲) embedded in a representation space. (b) The clusters of points $\mathcal{C}$ obtained by DIRECTPROBE. Each cluster has only one label, but there are two ▲ clusters. (c) Ten interpolated data points (●). The original dataset is also shown in the panel for reference, but is not used as METAPROBE training data. (d) The N classifiers that are trained on the original data. (e) The classifiers are applied to the sampled points to obtain the predicted label distribution, which is the supervision for training METAPROBE.

Algorithm 1 Training METAPROBE using a collection of dataset-representation pairs $\mathcal{D}$.

Input: Dataset-Rep pairs $\mathcal{D} = \{(D, \phi)\}$, METAPROBE feature extractor ψ, Number of task-classifiers N
Output: METAPROBE model W

```
Initialize data for METAPROBE D_g ← ∅
for (D, φ) ∈ D do
    C ← DIRECTPROBE(D, φ)
    H ← Train N classifiers on (D, φ)
    D̂ ← generate interpolated data from D, φ, H
    D_g ← D_g ∪ ψ(D̂, C)
end for
Train a linear model W on D_g
return W
```

Representations. We conduct experiments on four English representations: RoBERTa$_{\text{base}}$, RoBERTa$_{\text{large}}$ (Liu et al., 2019b), DistilBERT (Sanh et al., 2019) and ELMo (Peters et al., 2018). As a group, they represent a diverse collection of architectures, pre-training methods, and pre-training data. Their characteristics are summarized in Appendix B.

Tasks. We investigate five different English NLP tasks represented by ten datasets, covering various usages of word representations, and diverse linguistic phenomena. We briefly describe them here and Appendix A gives details about these tasks.

Preposition supersense disambiguation (PS) is a task of predicting supersense labels for single-token prepositions. It involves two sets of labels: **semantic role (PS-role)** and **semantic functions (PS-func)**. We use the annotation from Streusle v4.2 corpus (Schneider et al., 2017).

Part-of-speech Tagging (POS) the task of predicting part-of-speech tags for each token in the sentence. We use the English portion of the parallel universal dependencies treebank (ud-pud, Nivre et al., 2016).

Groups	Tasks	Datasets	Representations
Training	PS,POS,TC	PS-role, PS-func, POS TREC-6, TREC-50	RoBERTa$_{\text{base}}$ RoBERTa$_{\text{large}}$
Test	NER,SR,TC	CoNLL04, SciERC SemEval, ATIS	ELMo DistilBERT

Table 1: Summary of datasets and representations.

Text Classification (TC) predicts a label for a piece of text. We use three datasets in this work: **TREC-6**, **TREC-50** (Li and Roth, 2002) and **ATIS** (Tür et al., 2010).

Semantic Relation (SR) involves predicting the semantic relation between a pair of nominals. We use three datasets for semantic relation task: **CoNLL04** (Roth and Yih, 2004), **SciERC** (Luan et al., 2018) and **SemEval 2010 Task 8 (SemEval)** (Hendrickx et al., 2010) dataset.

Named Entity Recognition (NER) requires finding named entities in a sentence. In this work, we use the **SciERC** (Luan et al., 2018) dataset.

The above datasets and representations are divided into training and test groups (shown in Table 1), which will be used in cross task/representations experiments in §5.2.

5 Experiments and Analysis

In this section, we will look at the main findings obtained using METAPROBE. Across our experiments, we compare the performance of METAPROBE against a linear probe and a nearest neighbor probe (1-NN). The linear probe baseline represents a general setting of predictive probing.

The 1-NN probe builds upon the simplest geometric property: the distance between one unseen example and its closest labeled example. For efficiency, for METAPROBE, we consider the closest $c = 10$ clusters of unseen examples.[6] To account for outliers, we ignore clusters with fewer than two examples.

Note that we do not seek to build a better probe. Instead, METAPROBE provides a comparable performance and also exposes the more sophisticated geometric structures of the representations used by predictors, which other probing approaches that operate directly on embeddings can not reveal.

5.1 Single Task & Representation

First, let us examine if the geometry of representations can predict the task-classifiers' behavior. Table 2 summarizes the results for single task/representation setting, where we evaluate METAPROBE on the test sets using the representations on which it was trained. We make the prediction using the label with maximum probability predicted by METAPROBE.

Finding 1: The geometry of representations indeed contains information that can predict the decisions of task-classifiers. The third column from the left of Table 2 shows the accuracies of METAPROBE. The last three columns show the accuracy differences between METAPROBE and a linear probe, 1-NN probe, and average accuracy over 1010 task-classifiers respectively. For example, a linear probe achieves 79.65% (77.90+1.75) with $\text{RoBERTa}_{\text{base}}$ on the PS/role task. It is not surprising that the linear probe sometimes fares better than METAPROBE; the latter is trained for the *predicted label distribution* (instead of gold labels) and uses the sampled points (instead of the actual training set). What is surprising is that METAPROBE matches or even outperforms a linear probe in some cases. We see that METAPROBE achieves higher accuracy on four of ten cases than a linear probe and 1-NN probe, and on three of ten cases than the average accuracy over 1010 task-classifiers. These observations suggest that using *only* geometric features from the embeddings, METAPROBE can predict task-classifiers' predictions.

[6]We consider the ten closest DIRECTPROBE clusters to keep computation costs down. Importantly, this does not change the results; an example is unlikely to be merged with distant clusters. In terms of the implementation, this choice means that the matrix M in Equation (3) is a $\{0,1\}^{c\times m}$ matrix, and that each example x has its own separate M because different examples will have different sets of closest clusters.

Task/Dataset	Rep	MetaProbe	Linear Probe	1-kNN Probe	Average CLS
PS/role	$\text{RoBERTa}_{\text{base}}$	77.90	+1.75	+0.66	+1.23
	$\text{RoBERTa}_{\text{large}}$	74.18	+3.06	0.00	+1.01
PS/func	$\text{RoBERTa}_{\text{base}}$	83.15	+5.69	+5.03	+5.14
	$\text{RoBERTa}_{\text{large}}$	82.71	+3.72	+1.31	+3.67
POS/ud-pud	$\text{RoBERTa}_{\text{base}}$	94.06	+0.37	+0.21	+0.34
	$\text{RoBERTa}_{\text{large}}$	93.85	-0.37	-0.44	-0.11
TC/TREC-6	$\text{RoBERTa}_{\text{base}}$	91.40	-3.20	+0.60	+0.79
	$\text{RoBERTa}_{\text{large}}$	90.80	-1.20	-0.20	-0.43
TC/TREC-50	$\text{RoBERTa}_{\text{base}}$	84.00	-7.20	-7.00	-0.93
	$\text{RoBERTa}_{\text{large}}$	81.40	1.80	-3.00	1.08

Table 2: Accuracy comparsion in single task/representation setting. The last three columns show the accuracy difference between METAPROBE and other probes and task-classifiers.

Finding 2: Different datasets and representations independently learned similar patterns. Since METAPROBE is a linear model, we can examine its weights to see if the task-classifiers trained on different representations and datasets exploit similar geometric properties. To do so, we compute the Pearson correlation between the METAPROBE weights learned from different datasets and representations. The correlations range from 0.58 to 0.97, averaging at 0.80, with even the lower end being a strong positive correlation. Appendix G shows the full table. The high correlations of these feature weights suggest that the patterns behind the decision process of task-classifiers are universal.

5.2 Cross Tasks & Representations

We hypothesized above that task-classifiers make decisions based on universal geometric patterns. A natural experiment to verify this hypothesis is to apply the learned model to unseen datasets and representations. To this end, in this subsection, we investigate the cross tasks/representations setting.

We sampled $2k$ training points from each dataset and representation in the training group of Table 1 to compose a cross task/representation training set (with $20k$ points in all). We train METAPROBE on this cross task/representation dataset and evaluate it on unseen tasks (test tasks in Table 1) and representations. If the patterns identified by METAPROBE are not stable across tasks and representations, then we expect that METAPROBE show much lower accuracies than the other probes.

Finding 3: There exist universal geometric patterns across tasks and representations. Table 3 shows the results of cross tasks/representations setting. We first observe that METAPROBE shows higher accuracy on nine out of ten cases than a

Task/Dataset	Rep	MetaProbe	Linear Probe	1-kNN Probe	Average CLS
NER/SciERC	elmo	76.62	+0.95	-0.95	-2.08
	DistilBERT	74.96	+3.50	-1.78	+2.86
SR/CoNLL04	elmo	96.45	0.00	+0.47	+0.20
	DistilBERT	94.31	+0.47	-6.64	+0.98
SR/SciERC	elmo	74.64	+1.85	-3.49	+0.20
	DistilBERT	77.10	+2.16	-8.32	+0.13
SR/SemEval	elmo	74.35	+1.91	-4.05	+2.01
	DistilBERT	75.52	+3.46	-4.01	+2.10
TC/ATIS	elmo	95.07	-1.46	-4.59	-0.27
	DistilBERT	93.39	-2.35	-2.80	-0.60

Table 3: Accuracy comparsion in cross task/representation setting. Note that METAPROBE is trained from the datasets and representations in the training group. All the accuracies are evaluated based on the same METAPROBE parameters. On the other hand, other probes and classifiers are trained individually.

1-NN probe, on two out of ten cases than a linear probe, and on three out of ten cases than average task-classifiers. Note that each linear probe, 1-NN probe, and task-classifier is trained individually for each task/representation while the METAPROBE parameters are unchanged. Results in Table 3 are surprising because recall that we are operating in the cross tasks/representation setting, where METAPROBE never sees these tasks (e.g. NER and Semantic Relation) and representations (e.g. ELMo and DistilBERT) during its training stage. Thus, we do not expect METAPROBE would achieve good performances. Nevertheless, it predicts with high accuracy for unseen datasets and representations. These observations show that METAPROBE captures universal geometric patterns of the decision process of task-classifiers.

Finding 4: Different tasks/representations can use geometry differently. To understand how different features affect these universal patterns, we run ablations on the three groups of features from §3.2. Table 4 shows partial results (A full analysis can be found in Appendix E). We observe that different sets of features contribute differently based on the tasks and datasets. For example, SR/SciERC relies heavily on cluster-only features; NER/SciERC relies on the distance features, while SR/SemEval relies on the merging features. These observations suggest that although some universal patterns exist, different datasets and representations still have rely on different geometric structures.

5.3 Analyzing METAPROBE's Weights

In this subsection, we show three features with distinguishable weights from cross task/representation settings as examples to illustrate how geometry affects task-classifiers' decisions. In Appendix H, we show the list of features with top 10 absolute weights.

Dataset	Rep	Cluster-only	W/o Distance	Merging
NER/SciERC	ELMo	+2.08	-8.55	-0.42
	DistilBERT	+2.02	-4.81	-3.03
SR/SciERC	ELMo	-5.65	-1.95	-2.87
	DistilBERT	-6.98	-3.08	-8.11
SR/SemEval	ELMo	-0.15	-2.32	-3.35
	DistilBERT	-0.33	-1.69	-8.35

Table 4: Ablations in cross task/representation settings. The numbers in the table are the difference of accuracy against the full features (shown in Table 3). A full ablation table can be found in Appendix E.

x^{cdis} is a feature that quantifies the distance between a new example and the centroid of a cluster, and is defined such that a closer distance means a larger x^{cdis} value. For example, in Figure 1 (left), the example represented by the $\star$ will have a higher value for this feature for cluster C_i than C_j. After training, we observe that x^{cdis} has one of the largest positive values, suggesting that a task-classifier tends to predict the label of closer clusters for the unseen example, which is intuitive.

x_j^{mhd} is a set of features that quantify how much of the max-margin hyperplane between current cluster C_i and j-th closest cluster would change if we merge the example with C_i (see Figure 1). After training, we see that x_j^{mhd} has the smallest negative value, suggesting that if merging the example to a cluster C_i causes a large change to the layout, the classifier tends to *not* make this decision even if current unseen example is close to C_i. This is not captured by the neighborhood-only baseline.

$c^{average}$ is a feature to describe the properties of the cluster under consideration. It is the average pairwise distance inside the cluster. After training, we find that $c^{average}$ has a weight close to zero, suggesting that this feature does not contribute to the decisions of task-classifiers. It shows that the properties of the cluster itself do not affect the predictions. Instead, the geometric relations between the unseen example and clusters (e.g. x^{cdis} and x_j^{mhd}) have more impact on the decisions of task-classifiers.

6 Conclusions

A long list of probing work involves examining the predictiveness of linguistic properties, both

syntax (e.g., Kassner and Schütze, 2020) and semantic (e.g., Aghajanyan et al., 2021). To ease the uncertainty of training classifiers, Hewitt and Liang (2019); Senel et al. (2018) propose to use controlled test sets to verify if the representations encode meaningful linguistic information. Other work (e.g. Marvin and Linzen, 2018; Wu et al., 2020) proposed error analysis to reverse engineer the information encoded in representations.

In this paper, we ask: *Are there universal geometric patterns in contextualized representations that can explain the decisions of task-classifiers?* We answer the question by developing METAPROBE, a linear probe that predicts trained classifier's predictions using the geometric features of the representations. Via experiments, we verify the existence of universal patterns in contextualized representations that models exploit. The patterns learned by METAPROBE can be used to make predictions on unseen tasks and representations. Finally, by analyzing the learned weights, we show how geometric properties affect the decisions of task-classifiers.

7 Limitations

We rely on the off-the-shelf implementation of DIRECTPROBE to extract features. However, DIRECTPROBE is limited in its scalability for large datasets. For example, more than $20k$ training examples makes it extremely slow. (We expect that tool itself may be made faster with engineering efforts, but that was not our focus.) As a result, our feature extraction strategy suffers from the same problem.

Acknowledgements

This work is partially supported by NSF grants #1801446, #1822877, #2007398 and #2129111. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies of any government agency.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. 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 7319–7328, Online. Association for Computational Linguistics.

Ehsan Aghazadeh, Mohsen Fayyaz, and Yadollah Yaghoobzadeh. 2022. Metaphors in pre-trained language models: Probing and generalization across datasets and languages. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2037–2050, Dublin, Ireland. Association for Computational Linguistics.

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. 2022. Probing for constituency structure in neural language models. *CoRR*, abs/2204.06201.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Comput. Linguistics*, 48(1):207–219.

Tyler A Chang, Zhuowen Tu, and Benjamin K Bergen. 2022. The Geometry of Multilingual Language Model Representations. *arXiv preprint arXiv:2205.10964*.

Shivani Choudhary, Niladri Chatterjee, and Subir Kumar Saha. 2022. Interpretation of black box NLP models: A survey. *CoRR*, abs/2203.17081.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

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 the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

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.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. 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 55–65,

Hong Kong, China. Association for Computational Linguistics.

Utku Evci, Vincent Dumoulin, Hugo Larochelle, and Michael C. Mozer. 2022. Head2toe: Utilizing intermediate representations for better transfer learning. *CoRR*, abs/2201.03529.

Hila Gonen, Shauli Ravfogel, and Yoav Goldberg. 2022. Analyzing gender representation in multilingual models. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 67–77, Dublin, Ireland. Association for Computational Linguistics.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *International Conference on Learned Representations (ICLR)*.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, Uppsala, Sweden. 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.

Dieuwke Hupkes and Willem H. Zuidema. 2018. Visualisation and 'diagnostic classifiers' reveal how recurrent and recursive neural networks process hierarchical structure (extended abstract). In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5617–5621. ijcai.org.

Alexander Immer, Lucas Torroba Hennigen, Vincent Fortuin, and Ryan Cotterell. 2022. Probing as quantifying inductive bias. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1839–1851, Dublin, Ireland. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Najoung Kim, Roma Patel, Adam Poliak, Patrick Xia, Alex Wang, Tom McCoy, Ian Tenney, Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, and Ellie Pavlick. 2019. Probing what different NLP tasks teach machines about function word comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.

Karim Lasri, Tiago Pimentel, Alessandro Lenci, Thierry Poibeau, and Ryan Cotterell. 2022. Probing for the usage of grammatical number. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8818–8831, Dublin, Ireland. Association for Computational Linguistics.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

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.

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. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Julian Michael, Jan A. Botha, and Ian Tenney. 2020. Asking without telling: Exploring latent ontologies

in contextual representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6792–6812, Online. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

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.

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, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020b. 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.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8592–8600.

Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. 2019. Classification and clustering of arguments with contextualized word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 567–578, Florence, Italy. Association for Computational Linguistics.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Naomi Saphra and Adam Lopez. 2019. Understanding learning dynamics of language models with SVCCA. 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 3257–3267, Minneapolis, Minnesota. Association for Computational Linguistics.

Nathan Schneider, Jena D Hwang, Archna Bhatia, Vivek Srikumar, Na-Rae Han, Tim O'Gorman, Sarah R Moeller, Omri Abend, Adi Shalev, Austin Blodgett, et al. 2017. Adposition and case supersenses v2. 5: Guidelines for english. *arXiv e-prints*, pages arXiv–1704.

Lutfi Kerem Senel, Ihsan Utlu, Veysel Yücesoy, Aykut Koç, and Tolga Çukur. 2018. Semantic structure and interpretability of word embeddings. *IEEE ACM Trans. Audio Speech Lang. Process.*, 26(10):1769–1779.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

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. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mycal Tucker, Tiwalayo Eisape, Peng Qian, Roger Levy, and Julie Shah. 2022. When does syntax mediate neural language model performance? evidence from dropout probes. *CoRR*, abs/2204.09722.

Gökhan Tür, Dilek Hakkani-Tür, and Larry P. Heck. 2010. What is left to be understood in atis? In *2010 IEEE Spoken Language Technology Workshop, SLT 2010, Berkeley, California, USA, December 12-15, 2010*, pages 19–24. IEEE.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, Florence, Italy. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2019. The bottom-up evolution of representations in the transformer: A study with machine translation and language modeling objectives. 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 4396–4406, Hong Kong, China. Association for Computational Linguistics.

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.

Zi Wang, Alexander Ku, Jason Baldridge, Thomas L Griffiths, and Been Kim. 2023. Gaussian process probes (gpp) for uncertainty-aware probing. *arXiv preprint arXiv:2305.18213*.

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.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176, Online. Association for Computational Linguistics.

Jiannan Xiang, Huayang Li, Defu Lian, Guoping Huang, Taro Watanabe, and Lemao Liu. 2022. Visualizing the relationship between encoded linguistic information and task performance. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 410–422, Dublin, Ireland. Association for Computational Linguistics.

Christos Xypolopoulos, Antoine Tixier, and Michalis Vazirgiannis. 2021. Unsupervised word polysemy quantification with multiresolution grids of contextual embeddings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3391–3401, Online. Association for Computational Linguistics.

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.

Yichu Zhou and Vivek Srikumar. 2022. A closer look at how fine-tuning changes BERT. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1046–1061, Dublin, Ireland. Association for Computational Linguistics.

	Task/Dataset	#Train	#Test	#Label	Rep Type
Training	PS/Role	4282	457	47	Token
	PS/Func	4282	457	40	Token
	POS/ud-pud	16,860	4323	17	Token
	TC/TREC-50	5452	500	50	Sentence
	TC/TREC-6	5452	500	6	Sentence
Test	NER/SciERC	5598	1685	6	Span
	SR/ConLL04	1283	422	5	Span-pair
	SR/SciERC	3219	974	7	Span-pair
	SR/SemEval	8000	2717	19	Span
	TC/ATIS	4978	893	26	Sentence

Table 5: A summary of tasks and datasets in our experiments.

A Summary of Datasets

In this work, we conduct experiments on five tasks using ten different English language datasets. Table 5 shows the statistics of these tasks and datasets.

We obtain PS/Role and PS/Funcs datasets from the annotation of Streusle v4.2 corpus (Schneider et al., 2017), which can be downloaded from `https://github.com/nert-nlp/sterile`. We download the POS/ud-pud dataset from `https://github.com/UniversalDependencies/UD_English-PUD`. TREC dataset is downloaded from `https://cogcomp.seas.upenn.edu/Data/QA/QC/`. TC/ATIS dataset is obtained from `https://github.com/howl-anderson/ATIS_dataset/blob/master/README.en-US.md`. We obtain the CoNLL04 and SciERC datasets using the scripts from `https://github.com/lavis-nlp/spert/`. The SemEval dataset is downloaded from `http://www.kozareva.com/downloads.html`

For tasks that require sentence embeddings, we use the first token (`<s>` or `[CLS]`) to represent sentences for $\text{RoBERTa}_{\text{base}}$, $\text{RoBERTa}_{\text{large}}$, and DistilBERT. For ELMo, we average the embeddings of all tokens in the sentence. We use the average embedding of the tokens within the span for the phrases with more than one token.

B Summary of Representations

Table 6 shows the statistics of the representations we used in this work. We use the original implementation of ELMo and the HuggingFace library (Wolf et al., 2020) for the others.

Grouping	Rep	#Param	Pre-training	Dim
Training	$\text{RoBERTa}_{\text{base}}$	125M	MLM	768
	$\text{RoBERTa}_{\text{large}}$	355M	MLM	1024
Test	DistilBERT	66M	distillation	768
	ELMo	94.6M	LM	1024

Table 6: Statistics of the four representations in our experiments.

Task/Dataset	Rep	Training Time (in hours)	Acc	STD
PS/Role	$\text{RoBERTa}_{\text{base}}$	4.5	79.13	1.00
	$\text{RoBERTa}_{\text{large}}$	5	75.19	1.48
PS/Func	$\text{RoBERTa}_{\text{base}}$	5	88.29	0.72
	$\text{RoBERTa}_{\text{large}}$	5.5	86.39	0.86
POS/ud-pud	$\text{RoBERTa}_{\text{base}}$	29	94.40	0.26
	$\text{RoBERTa}_{\text{large}}$	88	93.74	0.27
TC/TREC-50	$\text{RoBERTa}_{\text{base}}$	22.5	92.19	1.44
	$\text{RoBERTa}_{\text{large}}$	25	90.37	0.95
TC/TREC-6	$\text{RoBERTa}_{\text{base}}$	31	83.07	2.19
	$\text{RoBERTa}_{\text{large}}$	31.5	82.48	1.50
NER/SciERC	ELMo	2.5	74.54	1.71
	DistilBERT	3	77.81	1.03
SR/ConLL04	ELMo	0.5	96.65	0.56
	DistilBERT	0.5	95.29	0.63
SR/SciERC	ELMo	1	74.84	0.89
	DistilBERT	1.5	77.23	1.31
SR/SemEval	ELMo	2	76.36	0.68
	DistilBERT	2	77.63	0.77
TC/ATIS	ELMo	17.5	94.80	0.81
	DistilBERT	13	92.80	1.86

Table 7: Statistics of the 1010 task-classifiers. The last two columns shows the average and standard deviation of the accuracy on the test set.

C Summary of Task Classifiers

In our experiments, we collect the predictions from a large list of different classifiers. We use linear classifiers, one layer neural networks with hidden layer sizes of $(128, 256, 512, 1024)$ and two-layers neural networks with hidden layer sizes of $(128, 256, 512, 1024) \times (128, 256, 512, 1024)$. For the activations functions, we use Sigmoid, Tanh, ReLU, LeakyReLU and ELU (Clevert et al., 2016). All the neural networks are optimized by AdamW (Loshchilov and Hutter, 2019) with a learning rate of 0.001 and batch size of 128. We set the maximum iterations to be 50 for each run. We run ten classifiers using different initializations for each architecture. We use cross-entropy loss for all classifiers. In total, we have 1010 classifiers for each dataset and representation. We run our models on a single Titan GPU. Table 7 shows the training statistics of the 1010 task-classifiers for each dataset and representation.

D Optimization Details of METAPROBE

Since METAPROBE is a linear model, we optimize it by min-batch gradient descent with batch size of 128 and leanring rate of 0.001 for all experiments. We set the maximum iteration number to be 400. We choose these hyperparameters by tuning on an extra sampled dev set (described in §3.3). We run our models on a single Titan GPU.

E Ablation Study

Dataset	Rep	Cluster-only	W/o Distance	W/o Merging
NER/SciERC	ELMo	+2.08	-8.55	-0.42
	DistilBERT	+2.02	-4.81	-3.03
SR/CoNLL04	ELMo	0.00	+0.47	+0.47
	DistilBERT	-0.24	+0.95	-6.16
SR/SciERC	ELMo	-5.65	-1.95	-2.87
	DistilBERT	-6.98	-3.08	-8.11
SR/SemEval	ELMo	-0.15	-2.32	-3.35
	DistilBERT	-0.33	-1.69	-8.35
TC/ATIS	ELMo	-0.90	-1.23	-5.71
	DistilBERT	-1.01	-2.13	-41.10

Table 8: Ablations in cross task/representation settings. The numbers in the table are the difference of accuracy against the full features (shown in Table 3).

One interesting observation of Table 8 is DistilBERT on TC/ATIS whose accuracy decreases 40% after removing merging features. After analyzing the weights of ablated METAPROBE, we found that: (i) METAPROBE penalizes the clusters with large number of examples, i.e.METAPROBE does not want to assign unseen examples to large clusters; (ii) METAPROBE awards the small distance between unseen examples and clusters, i.e.METAPROBE prefers to assign unseen example to their closest clusters. The first observation follows the intuition that we do not want be biased by unbalanced labels. The second observation forms the basis of the nearest neighbors.

Since ATIS is a highly unbalanced dataset, when represented by DistilBERT, the majority unseen examples are closest to the largest cluster but not close enough to overrun the negative effect of large cluster. Thus, without merging features, METAPROBE chooses to merge these examples to the second closest cluster instead of the closest one. On the other hand, when represented by ELMo, the distances between the unseen examples and the closest clusters are close enough to overrun the effect of the size of cluster.

F Feature Template

Given a representation ϕ and a labeled dataset D, let $\mathcal{C} = \{C_i\}$ be the set of clusters returned by DIRECTPROBE. We use $C_i (1 \leq i \leq k)$ to denote the i-th closest cluster to the unseen example x in the representation. For eac pair of (x, C_i), we extract cluster-only feartures (shown in Table 9), distance feartures (shown in Table 10) and merging feartures (shown in Table 11).

G Correlations Between Weights

Table 12 shows the Pearson coefficient correlations between weights learned from different tasks and representations.

H Top-10 Weights

Table 13 shows the weights that have top 10 absolute values. These weights are trained from the cross-task-cross-representation setting.

I Robustness

Table 14 shows the average accuracy difference between METAPROBE and a linear probe when only using 1% to 10% of the training set. For each percent, we repeat the sampling for 50 rounds and apply a two-sided sample paired t-test. Except for the 1% and 10% case, all other percentages are statistically significant (with a p-value less than .0001).

Symbols	Description	#Features	Range	Notes
c^{num}	The proportion of examples inside cluster C_i	1	$[0, 1]$	
$c^{average}$	The average distance inside each cluster normalized by its maximum distance	1	$[0, 1]$	
c^{std}	The standard deviation of distances inside each cluster normalized by its maximum distance.	1	$[0, 1]$	
c_j^{dis}	The distances between C_i and C_j normalized by the maximum distance	k	$[0, 1]$	$j \in [1..k]$; When $i = j$,$c_j^{dis} = 0$.

Table 9: Cluster-only features for the i-th closest cluster C_i of unseen example x. C_i is the current cluster we are considering.

Symbols	Description	#Features	Range	Notes
$inside$	If x is inside of C_i	1	$\{0, 1\}$	This feature can be 1 only when $i = 0$
x^{dis}	The i-th element of a score vector noramlized by softmaxing over the negative distances between x and each cluster in $\mathcal{C}$.	1	$[0, 1]$	
x^{cdis}	The i-th elment of a score vector noramlized by softmaxing over the negative distances between x and the *centroid* of each cluster in $\mathcal{C}$.	1	$[0, 1]$	
x_j^{span}	The distance between x and the span of C_i and C_j normalized by the maximum distance over all pairs of clusters.	k	$[0, 1]$	

Table 10: Distance features for the example x and its i-th closest cluster C_i. This set of feature is used to describe the relations between x and C_i based on the distances.

Symbols	Description	#Features	Range	Notes
$x\_not\_mergable$	If x can be merged by C_i without breaking the linear separablability condition.	1	$\{0, 1\}$	$x\_not\_mergable = 1$ when x cannot be merged by C_i
x_j^{mdis}	The distance between C_i and C_j after merging x to C_i divided by the original distance beween C_i and C_j.	k	$[0, 1]$	$x_j^{mdis} = 0$ for $i = j$; $x_j^{mdis} = 1$ if $C_i \cup \{x\}$ overlaps C_j
x_j^{mcos}	The cosine distance bewtween the weights of max-margin hyperplanes that separate C_i and C_j before and after merging x to C_i.	k	$[0, 1]$	$x_j^{mcos} = 0$ for $i = j$, $x_j^{mcos} = 1$ if $C_i \cup \{x\}$ overlaps C_j
x_j^{mhd}	The L1 distance between the weights of max-margin hyperplanes that separate C_i and C_j before and after merging x to C_i normalized by the sum of L1 norm of these two weights.	k	$[0, 1]$	$x_j^{mhd} = 0$ for $i = j$, $x_j^{mhd} = 1$ if $C_i \cup \{x\}$ overlaps C_j
x_{max}^{mdis}	The maximum value of x_j^{mdis} for $j \in [1..k]$	1	$[0, 1]$	
x_{max}^{mcos}	The maximum value of x_j^{mcos} for $j \in [1..k]$	1	$[0, 1]$	
x_{max}^{mhd}	The maximum value of x_j^{mhd} for $j \in [1..k]$	1	$[0, 1]$	

Table 11: Merging features for example x and its i-th closest cluster C_i. These features are used to quantify how much geometry will be changed if we merge x to C_i.

		RoBERTa$_{\text{base}}$					RoBERTa$_{\text{large}}$				
		PS-role	PS-func	POS	TREC-6	TREC-50	PS-role	PS-func	POS	TREC-6	TREC-50
RoBERTa$_{\text{base}}$	PS-role	1.00	0.97	0.91	0.95	0.80	0.73	0.63	0.64	0.79	0.88
	PS-func	0.97	1.00	0.85	0.94	0.74	0.74	0.57	0.58	0.73	0.83
	POS	0.91	0.85	1.00	0.93	0.87	0.72	0.62	0.64	0.79	0.86
	TREC-6	0.95	0.94	0.93	1.00	0.75	0.68	0.58	0.59	0.68	0.81
	TREC-50	0.80	0.74	0.87	0.75	1.00	0.83	0.74	0.76	0.90	0.84
RoBERTa$_{\text{large}}$	PS-role	0.73	0.74	0.72	0.68	0.83	1.00	0.69	0.76	0.80	0.76
	PS-func	0.63	0.57	0.62	0.58	0.74	0.69	1.00	0.96	0.79	0.72
	POS	0.64	0.58	0.64	0.59	0.76	0.76	0.96	1.00	0.76	0.73
	TREC-6	0.79	0.73	0.79	0.68	0.90	0.80	0.79	0.76	1.00	0.89
	TREC-50	0.88	0.83	0.86	0.81	0.84	0.76	0.72	0.73	0.89	1.00

Table 12: Pearson correlation coefficients between the weights learned from different datasets and representations.

Name	x_0^{mhd}	x_1^{mhd}	x_{max}^{mhd}	x_2^{mhd}	x_3^{mhd}	$x\_not\_mergable$	x^{cdis}	x_4^{mhd}	x_5^{mhd}	x^{dis}
Value	-12.19	-8.33	-8.17	-6.44	-6.31	5.77	5.72	-4.93	-3.97	3.86

Table 13: The weights with top 10 absolute values. Appendix F describes the explanations of these features.

Percentages	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Acc Difference	0.2%	**0.7%**	**0.7%**	**0.9%**	**0.8%**	**0.8%**	**0.6%**	**0.6%**	**0.6%**	0.4%

Table 14: Average accuracy difference between METAPROBE and a linear probe. Bold numbers indicate a statistically significant ($p < .0001$) improvement in favor of METAPROBE using the paired t-test.