Quantifying the Task-Specific Information in Text-Based Classifications

Anonymous ACL submission

Abstract

Recently, neural natural language models have attained state-of-the-art performance on a wide variety of tasks, but the high performance can result from superficial, surface-level cues (Bender and Koller, 2020; Niven and Kao, 2020). These surface cues, as the "shortcuts" inherent in the datasets, do not contribute to the *task-specific information* (TSI) of the classification tasks. While it is essential to look at the model performance, it is also important to understand the datasets. In this paper, we consider this question: Apart from the information introduced by the shortcut features, how much task-specific information is required to classify a dataset? We formulate this quantity in an information-theoretic framework. While this quantity is hard to compute, we approximate it with a fast and stable method. TSI quantifies the amount of linguistic knowledge modulo a set of predefined shortcuts – that contributes to classifying a sample from each dataset. This framework allows us to compare across datasets, saying that, apart from a set of "shortcut features", classifying the Multi-NLI task involves around 0.4 nats more TSI than the Quora Question Pair.

1 Introduction

Neural natural language processing (NLP) models have attained state-of-the-art classification tasks, including natural language inference, sentiment analysis, and textual similarity (Devlin et al., 2019; Yang et al., 2019). What drives this performance? A popular argument is: neural models learn certain linguistic skills for these tasks, and their representations encode linguistic knowledge (Lakretz et al., 2019; Hewitt and Manning, 2019; Chen et al., 2019; Tenney et al., 2019; Jiang and de Marneffe, 2019; Zhu et al., 2020; Ettinger, 2020). How can neural models encode this linguistic knowledge? Alain and Bengio (2017) suggested that, by attending to datasets, neural NLP models gradually learn to preserve useful, task-specific information while discarding the rest. In this way, the task-specific information is "distilled" in the neural network models. There are many text-based classification tasks (e.g., Williams et al. (2018)), each of which requires some amount of linguistic information to classify that the neural networks distill along the way.

Quora example
S1: What can make Physics easy to learn?
S2: How can you make physics easy to learn?
Label: True (similar question)
Correct reason: They have very similar meanings.
Shortcut: They both contain can, to, and "?".

MNLI example
S1: You have access to the facts.
S2: The facts are accessible to you.
Label: Entailment
Correct reason: S1 entails S2.
Shortcut: They both contain the, to and you.

Figure 1: Classifiers can rely on "shortcut features" to reach the correct predictions, but this strategy is not generalizable, since the classifiers do not learn the real linguistic knowledge. Shortcut features, including the occurrence of punctuations (e.g., "?") and stopwords (e.g., can, the, to, you), are prevalent in datasets, but should not be part of the linguistic knowledge required to classify. We propose a method to quantify how much task-specific, shortcut-irrelevant information remains in the datasets.

The inquiry into the information regime of models leads to an appealing goal in explainable AI (Doran et al., 2018): to infer the amount of task-specific, linguistic knowledge required for a given task in information-theoretic terms. With this unified metric, we will be able to compare across text-based classification tasks. Typically, classification accuracy and loss are used for comparison. However, recent research showed that a low cross-entropy loss might result from the information that is correlative but not causative to the prediction tasks. This is the "shortcut learning" problem, and it happens in a wide variety of classification

tasks (McCoy et al., 2019; Geirhos et al., 2020; Niven and Kao, 2020; Misra et al., 2020; Stali and Iacobacci, 2020) – even in human cognition, where study participants figure out more accessible ways to solve testing tasks (Geirhos et al., 2020).

Figure 1 presents two examples of shortcuts, where we could make predictions based on shortcuts that are irrelevant to the linguistic knowledge of the tasks. Therefore, shortcuts constitute a gap between how much *is learned* and how much *should be learned* to classify the task. Following the motivations of recent causal analysis papers (e.g., Elazar et al. (2021); Pryzant et al. (2021)), we want to factor out the impact of the shortcuts while still quantifying the amount of information a neural network model needs to learn for a task.

This paper presents a framework to separate the surface-level shortcuts from the deeper information. We quantify the "task-specific information" (TSI) that is not part of the spurious correlations. TSI is hard to compute numerically, but we use a method based on a Bayesian formulation to approximate this quantity (§3). The computation only requires computing cross-entropy losses on a pair of classification tasks. We discuss the proper choice of configurations to compute the TSI (Secs. 5.1,5.2). Our method is stable across dataset sizes (§ 5.3), and is easier to compute than existing entropy estimators (§ 5.4).

Overall, the TSI framework quantifies the "linguistic knowledge" required to perform text-based classifications and further allows principled comparisons of the degrees of linguistic knowledge across a wide range of classification tasks. For example, the classification task in MNLI dataset (Williams et al., 2018) requires about 0.25 nats more TSI than the sentiment detection task with IMDB movie reviews (Maas et al., 2011), and around 0.4 nats more than the textual similarity detection task with the QQP dataset (Wang et al., 2019) (§ 5.5), given a fixed set of shortcuts.

2 Related Work

Our work is related to prior work in identifying and isolating spurious artifacts ("shortcuts") in text-based prediction tasks, probing language embeddings for various linguistic phenomena, and analyzing dataset statistics.

Shortcut learning Deep neural networks can overtly rely on superficial heuristics, which allows them to perform well on standard benchmarks but prohibits generalization to real-world scenarios. Geirhos et al. (2020) called this problem "shortcut learning" and referred to these heuristics as "shortcuts". On text-based classification datasets, shortcuts appear in the form of spurious statistical cues. These include the warrants for argument reasoning (Niven and Kao, 2020), syntax heuristics and lexical overlaps in natural language inference (McCoy et al., 2019), and relevant words ("semantic priming") (Misra et al., 2020). These spurious surface cues do not contribute to task-specific information.

By carefully constructing test sets that do not have these statistical cues and spurious associations, such shortcuts can be diagnosed (Glockner et al., 2018; Gardner et al., 2020). Kaushik et al. (2020) counterfactually augmented text snippets in several sentiment-classification datasets via crowdsourcing by applying minimal changes to the original text to flip the prediction label. Rosenman et al. (2020) used challenge sets to reveal the "learning by heuristics" problem in the relation extraction task. In contrast to our work, none of these prior works formulate the issue of shortcut learning using information theory. Another strategy to factor out known dataset biases is debiasing algorithms, such as the residual fitting algorithm (He et al., 2019).

Probing The probing literature inspires our approach to analyzing the information in neural language models. According to Alain and Bengio (2017), the task of probing asks, "is there any information about factor ____ in this part of the model?" Following this line, many subsequent papers queried the amount of knowledge from various parts of neural models. These included syntax-related (Lakretz et al., 2019; Hewitt and Manning, 2019), semantic-related (Tenney et al., 2019), and discourse-related information (Chen et al., 2019; Koto et al., 2021). Towards developing reliable probing methods, several papers proposed control mechanisms (Pimentel et al., 2020; Zhu and Rudzicz, 2020). With a collection of imperfect classifiers, we can combine to adjust for potential confounds. Our analyses are motivated by this idea, but we study the classification instead of the probing regime.

Understanding the datasets In machine learning and NLP literature, several works studied the "difficulty" of datasets (Blache and Rauzy, 2011; Gupta et al., 2014; Collins et al., 2018; Jain et al.,

2020), but they did not consider factoring out the impacts of shortcuts. D'Amour et al. (2020) framed the shortcut learning issue as an underspecification problem: There is not enough information in training set to distinguish between spurious artifacts and the inductive biases (or rather, the linguistic knowledge). Recently, researchers have analyzed the behavior of models on individual samples during training to diagnose datasets (Tu et al., 2020; Kumar et al., 2019). Han et al. (2020) used influence functions to identify influential training samples and characterize the artifacts in datasets. Swayamdipta et al. (2020) computed metrics of training dynamics of a model, i.e., the prediction confidence and variability, to map a "cartography" of the data samples. Warstadt et al. (2020) introduced a dataset to study linguistic feature learning versus generalization in the RoBERTa base model and considered a probing setup with a control task to investigate the inductive biases of a pretrained model at the fine-tuning time. Lovering et al. (2021) found that the extent that a feature influences a model's decisions is affected by the probing extractability and its co-occurrence rate with the label. These works have a common intuition: we should study the datasets to study the spurious correlation (shortcuts). We follow this line of research and quantify the information of shortcuts in the datasets.

Mutual information Our work is related to information theory formulations about machine learning. Steinke and Zakynthinou (2020) proposed a formulation of conditional mutual information that can be used to reason about the generalization properties of machine learning models. Empirically, our proposed method (using the difference of a pair of cross-entropy losses) echoes what Xu et al. (2020) defined as the "predictive $\mathcal{V}$-information". We derive TSI from a different perspective from the $\mathcal{V}$-information. We elaborate in §3. A concurrent work, O'Connor and Andreas (2021), uses $\mathcal{V}$-information to study the effects of each context feature independently. In contrast, we consider the features in an aggregate manner.

3 Learning Task-Specific Information

This section presents our framework to quantify the task-specific information.

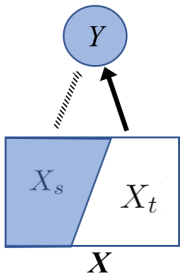

Figure 2: An illustration of the relationships between the text data X, containing a shortcut part X_s, and an unmeasurable task-specific part X_t, as well as the task label Y. The solid arrow indicates a causal relationship, while the dashed arrow indicates a spurious correlation. We want to factor out the observable X_s from this graph.

3.1 Removing the shortcuts

Consider a dataset of data points $\{(x_i, y_i)\}_{n=1}^N$, where $x_i \in \mathbb{R}^m$ is the feature vector, and y_i is the label. Let the random variable X represent all possible input features, and the random variable Y represent the task labels.

In our framework, the input random variable X constitutes of the shortcut part, denoted by a random variable X_s, and the task-specific part, an unmeasurable X_t. In other words, $X = f(X_s, X_t)$, where $X_s \perp\!\!\!\perp X_t$, and $f(\cdot)$ can be any composition function. Their dependency relationships can be described by Figure 2. This allows us to write the distributions as:

$$p(Y \mid X) = p(Y \mid X_t)p(Y \mid X_s) \underbrace{\frac{p(X_t)p(X_s)}{p(X)p(Y)}}_{\text{prior}} \tag{1}$$

When $X_s \perp\!\!\!\perp X_t$, $p(X) = p(X_t)p(X_s)$, so the prior term degenerates into $\frac{1}{p(Y)}$.

$$\begin{aligned} I(Y; X_t) &= \mathbb{E} \log \frac{p(Y, X_t)}{p(X_t)p(Y)} = \mathbb{E} \log \frac{p(Y \mid X_t)}{p(Y)} \\ &= \mathbb{E} \log \frac{1}{p(Y \mid X_s)} - \mathbb{E} \log \frac{1}{p(Y \mid X)} \\ &= H(Y \mid X_s) - H(Y \mid X) \end{aligned} \tag{2}$$

where the expectations are taken over the distribution implicitly defined by the data $\{x_i, y_i\}_{i=1}^N$. The equation in the second last line is acquired by substituting in Eq. 1.

3.2 Interpreting the model performance

Empirically, a model learning this task (e.g., a BERT (Devlin et al., 2019) with a fully connected layer on top) approximates the true, unknown distribution $p(Y \mid X)$. Let $q(Y \mid X)$ describe the learned model, then by definition:

$$H(Y \mid X) = \text{NLL}_{Y \mid X} - \text{KL}(p \,\|\, q) \quad (3)$$

where p and q are the short-hand notations of $p(Y \mid X)$ and $q(Y \mid X)$ respectively, and

$$\text{NLL}_{Y \mid X} = \mathbb{E}_{p(X)} \log \frac{1}{q(Y \mid X)} \quad (4)$$

is the cross-entropy loss. In this paper, we will use NLL to refer to the cross-entropy loss, for clarity.

A well-trained model would have high performance: a high accuracy, a low $\text{KL}(p \,\|\, q)$ divergence, and a low cross-entropy loss. However, as mentioned before, this could result from the model "taking shortcuts", predicting the task labels Y from the shortcuts X_s.

3.3 Computing TSI needs a control task

Here we consider a control task to specify the features that might benefit the classification but do not contribute to the linguistic knowledge required for the models to perform the task correctly. Figure 1 describes some shortcuts. We include the details in the Experiment below.

We refer to the classifier trained only on the shortcuts as the control model. When trained, the control model approximates the unknown distribution $p(Y \mid X_s)$ with an empirical distribution, $q(Y \mid X_s)$.

Definition 1: The *task-specific information* (TSI) in the classification task (described by X, Y) with respect to the shortcut X_s is quantified by:

$$I(Y; X_t) = \underbrace{\text{NLL}_{Y \mid X_s} - \text{NLL}_{Y|X}}_{\text{Known}} + \underbrace{\text{KL}(p_{Y \mid X} \,\|\, q_{Y \mid X}) - \text{KL}(p_{Y \mid X_s} \,\|\, q_{Y \mid X_s})}_{\text{Unknown}} \quad (5)$$

Similarly, $\text{NLL}_{Y \mid X_s}$ is the cross-entropy loss of the control task. They can be measured empirically, so we mark them as "known".

3.4 On the scales of the intractable KLs

In Eq. 5, the two "known" terms constitute of the predictive $\mathcal{V}$-information (Xu et al., 2020) from X_t to Y. Additionally, $I(Y; X_t)$ contains two intractable KL terms. As a sanity check, we use a collection of synthetic datasets to estimate their scales. Following are the distributions to generate these toy datasets $\{X, Y\}$:

$$X_j \sim \text{Bernoulli}(p_x), \text{ where } j \in \{1, 2, .., m\}$$
$$X = [X_1, X_2, ..., X_m]$$
$$Y = g(X_1, ..., X_m) + \epsilon, \text{ where } \epsilon \sim \text{Bernoulli}(p_y)$$

where m specifies the number of input features, and $g(X_1, ..., X_m)$ is a deterministic function. This construction allows an exact computation of the conditional entropy $H(Y \mid X)$. On the other hand, we compute the cross-entropy $\text{NLL}_{Y \mid X}$ by training a default scikit-learn MLPClassifier $q(Y \mid X)$ on the train portion of $\{X, Y\}$. Then, the difference between the dev loss and the conditional entropy is the KL values resulting from the imperfect classifier.

We generate toy datasets with different values of m ($2 \leq m \leq 10$), p_x and p_y (between 0.1 and 0.9). For $g(\cdot)$, we use two options:

- `sum`: $g(X) = \sum_j X_j$
- `and`: $g(X) = X_1 \wedge X_2 \wedge ... \wedge X_m$

Figure 3 show the histograms of the two options, respectively. In 99.5% (1184 of 1190) configurations, the dev losses are within 0.04 nats away from $H(Y \mid X)$. In other words, the scales of the $KL(p \,\|\, q)$ are estimated to be one magnitude smaller than those of $I(Y; X_t)$. In the subsequent analysis, we empirically ignore the intractable KL terms.

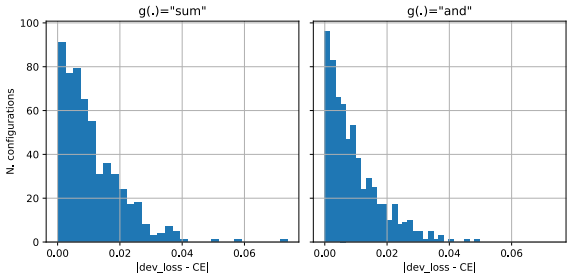

Figure 3: The histograms of $|NLL - H(Y \mid X)|$, i.e., the estimated scales of $\text{KL}(p \,\|\, q)$, with the `sum` and `and` option respectively.

3.5 Understanding TSI

Before moving to the computation, let us first briefly discuss some properties of TSI.

Lower bound. TSI ≥ 0, where equality is reached when the information from the shortcuts

(e.g., the presence of specific tokens) is sufficient for classification, so the model does not have to learn any task-specific knowledge to perform perfectly.

Upper bound. TSI $\leq H(Y)$, where the equality is reached when $H(Y \mid X_t) = 0$, i.e., the task label Y is a deterministic function of the task-specific variable X_t. Further, for a task with m distinct labels, Jensen-Shannon inequality gives us $H(Y) \leq \log m$ nats[1]. When $m = 2$ and 3, the TSI would be correspondingly upper-bounded by $\log 2 \approx 0.693$ and $\log 3 \approx 1.097$, respectively. When the number of classes m increases, the upper bound of TSI increases, resembling what Gupta et al. (2014) mentioned about how a larger number of classes contribute to the increased cross-entropy.

An on-average metric. TSI is averaged across the dataset samples, allowing comparison across datasets with different sizes. We can compare the TSI scores of a dataset with 50,000 samples (e.g., IMDB (Maas et al., 2011)) to that of a dataset with 400,000 samples (e.g., Quora Question Pairs) to directly compare their "linguistic informativeness". We discuss further about the dataset sizes in §5.3.

Quantity but not form. TSI quantifies the amount rather than describes the actual type of information required to classify a task. The former computes an aggregate metric, while the latter requires a deep understanding of the task knowledge. This paper considers the former.

4 Experiments

4.1 Datasets

We run experiments on several popular benchmarking datasets (in English) that test various linguistic abilities, including sentiment and attitude detection (Yelp and IMDB), entailment recognition (MNLI), and semantic similarity understanding (QQP). The dataset details are in Appendix A.

4.2 Control task features

The features for the control task need to be scalars. In the experiments, we use the following features to illustrate the application of our framework.

The occurrences of punctuations We count the punctuation in each input text sample and normalize by the number of tokens in the sentence. If a sample constitutes a pair of sentences, we concatenate the two sentences. Following is an example.

> You have access to the facts . The facts are accessible to you .

There are $N = 2$ occurrences of punctuations in the (concatenated) sentence with length $L = 14$, so the "occurrence of punctuation" feature is $\frac{2}{14}$.

The occurrence of stopwords We count the stopwords (modulo the negation words including "no", "nor", "don't" and "weren't") and normalize by the token length of the example. We concatenate the two sentences for the samples consisting of a pair of sentences similar to the punctuation feature. Following is an example.

> You have access to the facts . The facts are accessible to you .

There are $N = 8$ occurrences of stopwords in this sentence with length $L = 14$, so the "occurrence of stopword" feature is $\frac{8}{14}$. Note that some stopwords do have semantic roles. For example, `I`, `you` and `they` can specify the person(s) in the situations. Additionally, one could argue that the choice of stopwords between, e.g., `I` and `me` could indicate the role of the speaker, and so on. Therefore one could argue that the occurrence of stopwords can be a non-shortcut, dependent on the actual task. However, one can also argue for the opposite, since the information provided by these semantic roles seem irrelevant to various classification tasks – for example, both "I like this movie" and "You like this movie" would indicate a positive movie review. This collection serves as an example that the TSI framework allows considering a collection of semantically nontrivial words.

The overlapping of paired sentences For each pair of sentence (s_1, s_2), we use the number of overlapped tokens (relative to each of the two sentence lengths) to describe the extent of lexical overlapping. Following is an example.

> - What can make Physics easy to learn ?
> - How can you make Physics easy to learn ?

The two "lexical overlap" features for this sentence pair are overlap_1=$\frac{8}{9}$, overlap_2=$\frac{8}{10}$.

[1]Throughout this paper, we use nats (instead of bits) as the unit for measuring the information-theoretic terms.

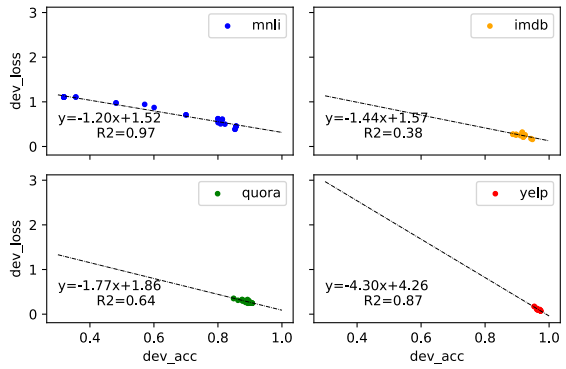

Figure 4: A scatter plot of the accuracy against dev loss of models trained on full datasets.

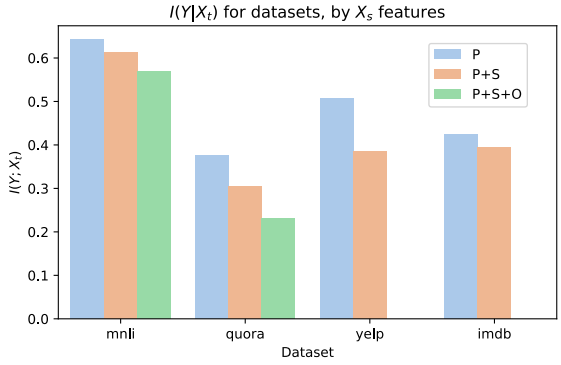

Figure 5: Estimates of TSI with different choices of shortcut features and the best models. Note that the `O` (lexical overlapping) heuristics only apply for MNLI and QQP, while the `P` (punctuation) and `S` (stopwords) heuristics apply to all four tasks. For each task, as more features are excluded, we can see the estimate decreases. Unless specifically mentioned, we consider $\text{TSI}^{\text{P+S}}$ for all tasks henceforth.

4.3 Classification models

For training $q(Y \mid X)$ models, we use BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2020), all on the base configuration (12 layers), with a fully connected head. Such transformer-based configurations are the state-of-the-art on classification tasks. We adopt the configurations of (Devlin et al., 2019): we concatenate the input sentences (for MNLI and QQP) and take the `[CLS]` token representations to pass in the fully connected head. The training hyperparameters follow the configurations recommended in the literature (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020; Wolf et al., 2019). For training $q(Y \mid X_s)$ models, we use MLPClassifier from scikit-learn (Pedregosa et al., 2011). We list the details in Appendix B.

5 Discussions

5.1 Estimating TSI with an suboptimal model

Each $\{X, X_s, Y\}$ configuration uniquely determines the $I(Y; X_t)$ value. Ideally, the models that perfectly fit the dataset distributions $p(Y|X)$ and $p(Y \mid X_s)$ can precisely estimate $I(Y\ ;\ X_t)$. Among all empirical models, the highest performing models approximate $I(Y; X_t)$ the most closely, since they lead to KL values (of Eq. 5) that are the smallest. Therefore, we report the results from finetuning the best of BERT, RoBERTa, and ALBERT, and we recommend using the best possible model.

Empirically, the model at hand might have an accuracy of several points lower than the top model at the GLUE leaderboard. How far do the entropy values of the imperfect models differ from those of the SOTA models (which usually only the accuracies are available)? Figure 4 plots the correlations between the cross-entropy losses and the accuracies of the non-degenerative finetuned $q(Y \mid X)$ models. Interestingly, except for IMDB, the results show linear trends, with the slopes and intercepts varying from task to task. The slopes of the trendlines could be used to interpolate the validation losses of the suboptimal models.

5.2 TSI and the choice of shortcuts

To enable cross-task comparisons, our framework considers TSI with respect to the fixed set of shortcuts. For example, apart from lexical overlap, how much linguistic information is there in classifying tasks? The choice of shortcut features affects the cross-entropy losses, hence the TSI.

Figure 5 reports the TSI estimations with various choices of shortcut features (additional results are in the Appendix). As we add features to the X_s set, $\text{NLL}_{Y \mid X_s}$ decreases, leading to a corresponding decrease in TSI. The lexical overlap feature exacerbates this decrease to X_s for MNLI. This follows our intuition since the syntactic heuristics such as lexical overlap have been identified as fallible heuristics for MNLI in prior work (McCoy et al., 2019), and though lexemes are shortcut features, they do encode semantics.

On the completeness of shortcuts. We do not aim at the unrealistic goal of exhausting all possible shortcuts. Instead, we present a framework where the contribution of the shortcuts, once identified, can be factored out. The TSI framework can generalize to additional shortcuts.

Generalization of features. We identified some

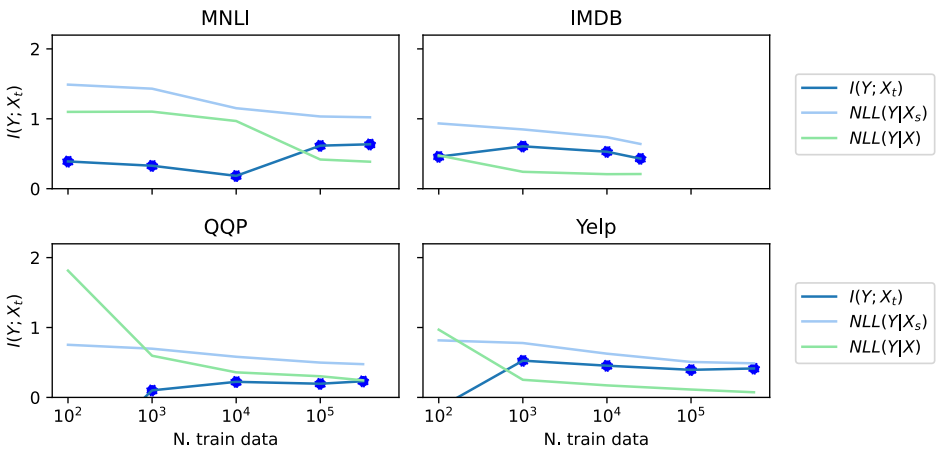

Figure 6: The $I(Y;X_t)$ estimation when we subsample different sizes of datasets.

features as "shortcut features". Dependent on the goal of analysis, one can apply other features (e.g., the length of sentences). In addition, automatic identification of shortcut features X_s method (e.g., approaches similar to those of Wang and Culotta (2020)) may be used as well.

5.3 How stable is TSI to dataset size?

To evaluate the effects of dataset size, we reduce the training sets with stratified sampling while assessing on the same validation set. As shown in Figure 6, the robustness of TSI estimations regarding the subset size differs across datasets. For MNLI, the estimation started to fluctuate starting from 25% of the original size. However, the estimates for IMDB, Quora, and Yelp remain relatively stable until we reduce the train set sizes to as few as $\sim 5\%$.

For both the $Y|X$ and $Y|X_s$ classification, the minimum reachable cross-entropy losses increases as we reduce the dataset sizes. A possible reason is that downsampling changes the data distribution and leads to mismatches between the train and the validation distributions. Similar effects are described in e.g., Gardner et al. (2020). Note that as we reduce the dataset sizes, $H(Y|X)$ rises faster than $H(Y|X_s)$, indicating that the deeper, task-specific knowledge requires more data to capture than those shortcut knowledge, echoing the finding of Warstadt et al. (2020).

5.4 What about alternative estimators?

Previous works have proposed several mutual information estimators based on setting up optimization goals, e.g., BA (Barber and Agakov, 2004), DV (Donsker and Varadhan, 1975), NWJ (Nguyen et al., 2010), MINE (Belghazi et al., 2018), CPC (Oord et al., 2018), and SMILE (Song and Ermon, 2020). We defer to Poole et al. (2019) and Guo et al. (2021) for summaries. Unfortunately, these variational methods do not directly apply to our problem setting. They involve modeling either the joint distribution $p(X,Y)$ or the generative distribution $p(X|Y)$. However, we consider the classification tasks where the state-of-the-art methods finetune the pretrained deep networks to model the conditional distributions of classification tasks $p(Y|X)$. It is possible to model the generative distribution on text classification datasets, but we consider that out of the scope of this paper. A recent paper, McAllester and Stratos (2020), argues in favor of using (and minimizing) the difference of entropies to estimate the terms related to mutual information because, unlike DV, NWJ, MINE, and CPC, this setting is not restricted to various statistical limitations.

How about directly estimating the entropy values $H(Y\,|\,X)$ and $H(Y\,|\,X_s)$ from data? It turns out that the computational effort required by this approach can easily grow prohibitive. Estimating the conditional entropy from the dataset $\{x_i, y_i\}_{i=1..N}$ involves finding the density, which is usually implemented by finding the nearest neighbors. This could require $\mathcal{O}(N\log N)$ computational time with $\mathcal{O}(N)$ memory[2] – where the memory requirements would grow prohibitively – or $\mathcal{O}(N^2)$ computational time with $\mathcal{O}(1)$ memory[3] – where the computational time would grow prohibitively. In com-

[2]Store all data points using a heap-like data structure, which allows query in $\mathcal{O}(\log N)$ time for each data point.

[3]Traverse the dataset to find nearest neighbors.

Dataset	$\text{Acc}_{Y \mid X}$	$\text{TSI}^{\text{P+S}}$	$\text{TSI}^{\text{P+S+O}}$
MNLI	0.85	0.68	0.64
IMDB	0.92	0.43	–
Yelp	0.97	0.41	–
QQP	0.89	0.31	0.23

Table 1: Our best estimates of TSI with P+S and P+S+O shortcut features respectively, and the dev accuracies of the corresponding $Y|X$ classifications.

parison, training two models with stochastic gradient descent requires only $\mathcal{O}(N)$ training time and $\mathcal{O}(1)$ memory. In other words, our method is more realistic under real-world computational constraints.

We run Monte Carlo simulations on a fraction of data using an off-the-shelf entropy estimator, NPEET (Kraskov et al., 2004). The sizes of the fractions are decided to be stable following the analysis of §5.3, i.e., 10^3 for IMDB and Yelp, 10^4 for Quora, and 10^5 for MNLI. We sample the subsets in a stratified manner with ten different random seeds. The conditional entropies $H(Y|X)$ and $H(Y|X_s)$ from Monte Carlo simulations differ significantly from those cross-entropy losses. Moreover, these simulations sometimes report negative $I(Y \mid X_t)$ values, indicating the prohibitive levels of the errors. We include the details in the Supplementary Data.

5.5 TSI required to classify each dataset

Table 1 contains our best estimations for TSI across datasets. The TSI^{P+S} of IMDB and Yelp are similar. Moreover, both TSI^{P+S} and TSI^{P+S+O} of MNLI are about 0.4 nats larger than those of QQP. Considering that the highest dev accuracy on MNLI and QQP are similar, the contrast in TSI provides an alternative perspective in comparing across tasks. When classifying the QQP dataset, neural models rely more on the artifacts, including punctuations and stopwords, than classifying the MNLI dataset.

Our method does not directly apply to HANS (McCoy et al., 2019) yet, since existing high-performing models mostly use HANS as a test set (e.g., He et al. (2019)). Instead of directly approximating the TSI of HANS, one can compute that of, e.g., HANS + MNLI.

5.6 Broader impacts

While there is a general momentum to develop better models on miscellaneous classification tasks, we call for more systematic comparisons across different datasets and propose developing datasets with higher "signal-to-noise ratios", as measured by, e.g., TSI. We also encourage the NLP community to think about several closely related problems:

Identifying shortcut features. While the release of a new NLP dataset is often paired with strong baselines for the proposed task, we also encourage future researchers to identify potential shortcuts or spurious associations, which could occur either due to the data collection procedure or due to the nature of the task itself (e.g., as reported by Romanov and Shivade (2018) for natural language inference tasks).

Leaderboard practices. Currently, the leaderboard practices reward high classification performances. We recommend that NLP researchers build leaderboards that additionally incentivize the minimal use of shortcuts. A potential way to do this would be constructing multiple test sets (Glockner et al., 2018), testing for different parameters of concern – such as data efficiency, fairness, etc., – as identified by Ethayarajh and Jurafsky (2020).

Metrics for cross-task comparison. Consider reporting the performance on a unified scale of "task-specific informativeness", rather than relying on average model performance metrics (Collins et al., 2018). Designing metrics with grounds in linguistic knowledge is an interesting direction of future work.

6 Conclusion

We propose a framework to quantify the task-specific information (TSI) for classifying text-based datasets. Given a fixed collection of shortcut features, TSI quantifies the linguistic knowledge attributable to the classification target that is *independent* of the shortcut features. The quantification method is computable under limited resources and is relatively robust to the dataset sizes. Further, this framework allows comparison across classification tasks under a standardized setting. For example, apart from the effects of punctuations and the non-negation stopwords, MNLI involves around 2.2 times TSI as the Quora Question Pairs, in terms of nats per sample.

References

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

David Barber and Felix Agakov. 2004. The IM algorithm: a variational approach to information maximization. *Advances in neural information processing systems*, 16(320):201.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540. PMLR.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data. In *ACL*, pages 5185–5198, Online. Association for Computational Linguistics (ACL).

Philippe Blache and Stéphane Rauzy. 2011. Predicting linguistic difficulty by means of a morpho-syntactic probabilistic model. In *Proceedings of PACLIC*, pages 160–167.

Mingda Chen, Zewei Chu, and Kevin Gimpel. 2019. Evaluation Benchmarks and Learning Criteria for Discourse-Aware Sentence Representations. In *EMNLP*, pages 649–662, Stroudsburg, PA, USA. Association for Computational Linguistics.

Edward Collins, Nikolai Rozanov, and Bingbing Zhang. 2018. Evolutionary data measures: Understanding the difficulty of text classification tasks. In *CoNLL*, pages 380–391. Association for Computational Linguistics.

Alexander D'Amour, Katherine Heller, Dan Moldovan, and et al. 2020. Underspecification in Machine Learning Underspecification Presents Challenges for Credibility in Modern Machine Learning. Technical report.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Monroe D Donsker and SR Srinivasa Varadhan. 1975. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28(1):1–47.

D Doran, SC Schulz, and TR Besold. 2018. What does explainable ai really mean? a new conceptualization of perspectives. In *CEUR Workshop Proceedings*, volume 2071.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic Probing: Behavioral Explanation with Amnesic Counterfactuals. *TACL*. ArXiv: 2006.00995.

Kawin Ethayarajh and Dan Jurafsky. 2020. Utility is in the eye of the user: A critique of NLP leaderboards. In *EMNLP*, pages 4846–4853, Online. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *TACL*, 8:34–48.

Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, et al. 2020. Evaluating NLP Models via Contrast Sets. *arXiv preprint arXiv:2004.02709*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut Learning in Deep Neural Networks. *arXiv e-prints*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Unintended cue learning: Lessons for deep learning from experimental psychology. *Journal of Vision*, 20(11):652–652.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *ACL*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Qing Guo, Junya Chen, Dong Wang, Yuewei Yang, Xinwei Deng, Lawrence Carin, Fan Li, and Chenyang Tao. 2021. Tight Mutual Information Estimation With Contrastive Fenchel-Legendre Optimization. *arXiv:2107.01131 [cs, math, stat]*. ArXiv: 2107.01131.

Maya R Gupta, Samy Bengio, and Jason Weston. 2014. Training Highly Multiclass Classifiers. *Journal of Machine Learning Research*, 15:1461–1492.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *ACL*, pages 5553–5563, Online. Association for Computational Linguistics.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn Dataset Bias in Natural Language Inference by Fitting the Residual. In *Workshop on Deep Learning Approaches for Low-Resource NLP*, pages 132–142. Association for Computational Linguistics.

John Hewitt and Christopher D Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *NAACL*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Abhinav Jain, Hima Patel, Lokesh Nagalapatti, Nitin Gupta, Sameep Mehta, Shanmukha Guttula, Shashank Mujumdar, Shazia Afzal, Ruhi Sharma

Mittal, and Vitobha Munigala. 2020. *Overview and Importance of Data Quality for Machine Learning Tasks*, pages 3561–3562. Association for Computing Machinery, New York, NY, USA.

Nanjiang Jiang and Marie-Catherine de Marneffe. 2019. Evaluating BERT for natural language inference: A case study on the CommitmentBank. In *EMNLP-IJCNLP*, pages 6088–6093. Association for Computational Linguistics.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. *ICLR*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Fajri Koto, Jey Han Lau, and Timothy Baldwin. 2021. Discourse probing of pretrained language models. In *NAACL*, pages 3849–3864, Online. Association for Computational Linguistics.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. 2004. Estimating mutual information. *Physical review E*, 69(6):066138.

Sachin Kumar, Shuly Wintner, Noah A Smith, and Yulia Tsvetkov. 2019. Topics to avoid: Demoting latent confounds in text classification. In *EMNLP-IJCNLP*, pages 4144–4154.

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. In *NAACL*, pages 11–20, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut, and Google Research. 2020. ALBERT: A Lite BERT For Self-Supervised Learning of Language Representations. In *ICLR*.

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

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

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 *ACL*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

David McAllester and Karl Stratos. 2020. Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 875–884. PMLR.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *ACL*, pages 3428–3448. Association for Computational Linguistics.

Kanishka Misra, Allyson Ettinger, and Julia Taylor Rayz. 2020. Exploring BERT's Sensitivity to Lexical Cues using Tests from Semantic Priming. In *Findings of EMNLP*.

XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. 2010. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861.

Timothy Niven and Hung Yu Kao. 2020. Probing neural network comprehension of natural language arguments. In *ACL*, pages 4658–4664. Association for Computational Linguistics (ACL).

Joe O'Connor and Jacob Andreas. 2021. What context features can transformer language models use? In *ACL*. Association for Computational Linguistics.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-Theoretic Probing for Linguistic Structure. In *ACL*. Association of Computational Linguistics.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. 2019. On Variational Bounds of Mutual Information. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 5171–5180. PMLR.

Reid Pryzant, Dallas Card, Dan Jurafsky, Victor Veitch, and Dhanya Sridhar. 2021. Causal effects of linguistic properties. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4095–4109, Online. Association for Computational Linguistics.

Alexey Romanov and Chaitanya Shivade. 2018. Lessons from natural language inference in the clinical domain. In *EMNLP*, pages 1586–1596. Association for Computational Linguistics.

Shachar Rosenman, Alon Jacovi, and Yoav Goldberg. 2020. Exposing Shallow Heuristics of Relation Extraction Models with Challenge Data. In *EMNLP*.

Jiaming Song and Stefano Ermon. 2020. Understanding the Limitations of Variational Mutual Information Estimators.

Ieva Stali and Ignacio Iacobacci. 2020. Compositional and Lexical Semantics in RoBERTa, BERT and DistilBERT: A Case Study on CoQA. In *EMNLP*.

Thomas Steinke and Lydia Zakynthinou. 2020. Reasoning About Generalization via Conditional Mutual Information. In *COLT*, volume 125 of *Proceedings of Machine Learning Research*, pages 3437–3452. PMLR.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, Yejin Choi, and Allen. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *EMNLP*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *ACL*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *TACL*.

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 *ICLR*.

Zhao Wang and Aron Culotta. 2020. Identifying spurious correlations for robust text classification. In *Findings of EMNLP*, pages 3431–3440, Online. Association for Computational Linguistics.

Alex Warstadt, Yian Zhang, Haau-Sing Li, Haokun Liu, and Samuel R Bowman. 2020. Learning Which Features Matter: RoBERTa Acquires a Preference for Linguistic Generalizations (Eventually). *EMNLP*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *NAACL*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

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

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*, pages 5753–5763.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NIPS*, pages 649–657.

Zining Zhu, Chuer Pan, Mohamed Abdalla, and Frank Rudzicz. 2020. Examining the rhetorical capacities of neural language models. In *EMNLP BlackboxNLP Workshop*, pages 16–32. Association for Computational Linguistics.

Zining Zhu and Frank Rudzicz. 2020. An information theoretic view on selecting linguistic probes. In *EMNLP*, pages 9251–9262, Online. Association for Computational Linguistics.

A Dataset details

- MNLI (Williams et al., 2018) contains 392.7k English sentence pairs as train set. MNLI evaluates whether a model can detect entailment relationships between those pairs. They provided two dev sets: the "matched" and the "mismatched" portion. We take the "matched" portion (with 9.8k sentence pairs) as the dev set, since they are derived from the same sources as the sentences in the training set.
- IMDB (Maas et al., 2011) is a large-scale dataset used to test a model's ability to detect sentiment from text. There are 50,000 movie reviews in English from IMDB in this dataset, with the training and dev sets containing 25,000 each.
- Yelp Reviews Polarity (Zhang et al., 2015) contains 560k and 38k (in training and dev portion respectively) customer reviews in English from Yelp. These are collected to decide the polarity of opinions.
- Quora Question Pairs[4] contains 404k English question pairs on Quora, created to test the abilities of the models to understand the semantics from text, and determine whether the question pairs are synonymous. We randomly divide the train-dev-test data with 80-10-10 portions (with numpy random permutation, seed 0).

B Hyperparameters

Following list the search space of our hyperparameters for modeling $Y|X$.

- Optimizer: We use Adam optimizer (Kingma and Ba, 2014) to train the model parameters, and use the initial learning rate of lr∈{2e-5, 1e-5}.
- Train epochs: For full datasets, we run 3 epochs. For training subsets with $N \in \{10^5, 10^4, 10^3\}$ samples, we run either 3 or 10 epochs. For training the small $N = 100$ sample subsets, we run $\{3, 10, 100\}$ epochs.
- Batch size: We run with batch sizes of $B \in \{2, 4, 8, 16\}$ for each classification setting. We find that in general, larger per-device batch sizes (e.g., 8 and 16) are better than smaller batches (e.g., 2 and 4), but a batch size of 16 or 32 could lead to out-of-memory issues on machines with 64GB memory.

[4] https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

Following the training procedure, our best development accuracies are comparable to the results reported on, e.g., the GLUE Benchmark leaderboard. While previous work added additional steps (e.g., learning rate warmup) to boost accuracy, our aim is not to beat the SOTA, but to establish a principled method that allows cross-task comparison. We include the hyperparameter configurations of all runs in the Supplementary Data.

For modelling $Y \mid X_s$, we use the scikit-learn (Pedregosa et al., 2011) MLPClassifier with hidden sizes from {10, 30, 100, 300, 10-10, 30-30, 100-100} where, e.g., 10-10 indicates two hidden layers with 10 units each. We rely on the default training procedures, search for the optimal hidden sizes based on the validation losses, and report the dev loss NLL$(Y \mid X_s)$ scores.