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.

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...
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 crowd-sourcing 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 (Blanche and Rauzy, 2011; Gupta et al., 2014; Collins et al., 2018; Jain et al., 2020).
206), 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. Levering 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.](image)

#### 3.1 Removing the shortcuts

Consider a dataset of data points $\{(x_i, y_i)\}_{i=1}^N$, 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 $X_s \perp X_t$, 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:

$$p(Y \mid X) = p(Y \mid X_t)p(Y \mid X_s) \frac{p(X_t)p(X_s)}{p(X)p(Y)}$$  \hspace{1cm} (1)

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

$$I(Y; X_t) = \mathbb{E} \log \frac{p(Y, X_t)}{p(X_t)p(Y)} = \mathbb{E} \log \frac{p(Y \mid X_t)}{p(Y)} = \mathbb{E} \log \frac{1}{p(Y \mid X_t)} - \mathbb{E} \log \frac{1}{p(Y \mid X)} = H(Y \mid X_s) - H(Y \mid X)$$  \hspace{1cm} (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:

$$H(Y \mid X) = \text{NLL}_{Y \mid X} - \text{KL}(p \parallel q)$$  \hspace{1cm} (3)

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

$$\text{NLL}_{Y \mid X} = E_{p(X)} \log \frac{1}{q(Y \mid X)}$$  \hspace{1cm} (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 KL$(p \parallel q)$ 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:

$$I(Y; X_t) = \underbrace{\text{NLL}_{Y \mid X_s} - \text{NLL}_{Y \mid X}}_{\text{Known}} + \underbrace{\text{KL}(p_{Y \mid X_s} \parallel q_{Y \mid X_s}) - \text{KL}(p_{Y \mid X} \parallel q_{Y \mid X_s})}_{\text{Unknown}}$$  \hspace{1cm} (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 $\{X, Y\}$:

$$X_j \sim \text{Bernoulli}(p_x), \text{ where } j \in \{1, 2, \ldots, m\}$$

$$X = [X_1, X_2, \ldots, X_m]$$

$$Y = g(X_1, \ldots, X_m) + \epsilon, \text{ where } \epsilon \sim \text{Bernoulli}(p_y)$$

where $m$ specifies the number of input features, and $g(X_1, \ldots, X_m)$ 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 NLL$_{Y \mid X}$ by training a default scikit-learn MLPClассifier $q(Y \mid X)$ 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:** $g(X) = X_1 \land X_2 \land \ldots \land X_m$

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 $KL(p \parallel q)$ 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.

3.5 Understanding TSI

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

**Lower bound.** TSI $\geq 0$, 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 \( H(Y) \leq \log m \) 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 compute 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.

<table>
<thead>
<tr>
<th>You have access to the facts. The facts are accessible to you.</th>
</tr>
</thead>
</table>

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.

<table>
<thead>
<tr>
<th>You have access to the facts. The facts are accessible to you.</th>
</tr>
</thead>
</table>

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 \( \text{overlap}_1 = \frac{2}{5} \), \( \text{overlap}_2 = \frac{2}{10} \).
ALBERT (Lan et al., 2020), all on the base con-

heuristics only apply for MNLI

TSI

vlin et al., 2019), RoBERTa (Liu et al., 2019), and

the details in Appendix B.

from scikit-learn (Pedregosa et al., 2011). We list

training

q

2019; Lan et al., 2020; Wolf et al., 2019). For

in the literature (Devlin et al., 2019; Liu et al.,

parameters follow the configurations recommended

the fully connected head. The training hyperpa-

take the

[CLS]

nate the input sentences (for MNLI and QQP) and

configurations of (Devlin et al., 2019): we concate-

of-the-art on classification tasks. We adopt the

Such transformer-based configurations are the state-

4.3 Classification models

For training \(q(Y \mid X)\) models, we use BERT (De-

vlin et al., 2019), RoBERTa (Liu et al., 2019), and

ALBERT (Lan et al., 2020), all on the base con-

figuration (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 concate-
nate the input sentences (for MNLI and QQP) and

take the \([CLS]\) token representations to pass in

the fully connected head. The training hyperpa-

rameters 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 deter-

mines the \(I(Y ; X_t)\) value. Ideally, the models

that perfectly fit the dataset distributions \(p(Y \mid X)\)

and \(p(Y \mid X_s)\) can precisely estimate \(I(Y ; X_t)\).

Among all empirical models, the highest perform-
ing 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 fine-
tuning 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 short-
cuts. 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 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

TSIP+SS for all tasks henceforth.

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. To enable cross-
task comparisons, our framework

considers TSI with respect to the fixed set of short-
cuts. 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.

5 Discussions

5.1 Estimating TSI with an suboptimal model

Each \(\{X, X_s, Y\}\) configuration uniquely deter-

mines the \(I(Y ; X_t)\) value. Ideally, the models

that perfectly fit the dataset distributions \(p(Y \mid X)\)

and \(p(Y \mid X_s)\) can precisely estimate \(I(Y ; X_t)\).

Among all empirical models, the highest perform-
ing 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 fine-
tuning 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 short-
cuts. 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 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

TSIP+SS for all tasks henceforth.

4.3 Classification models

For training \(q(Y \mid X)\) models, we use BERT (De-

vlin et al., 2019), RoBERTa (Liu et al., 2019), and

ALBERT (Lan et al., 2020), all on the base con-

figuration (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 concate-
nate the input sentences (for MNLI and QQP) and

take the \([CLS]\) token representations to pass in

the fully connected head. The training hyperpa-

rameters 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.
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 $Y \mid X$ and $Y \mid X_s$ 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, $H(Y \mid X)$ rises faster than $H(Y \mid X_s)$, 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 $p(X \mid Y)$. 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 $p(Y \mid X)$. 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 $H(Y \mid X)$ and $H(Y \mid X_s)$ 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 $O(N \log N)$ computational time with $O(N)$ memory$^2$ – where the memory requirements would grow prohibitively – or $O(N^2)$ computational time with $O(1)$ memory$^3$ – where the computational time would grow prohibitive. In com-

---

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

$^3$Traverse the dataset to find nearest neighbors.
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


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


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


10

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


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 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.


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

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 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 \(Y | X_s\), 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 \(\text{NLL}(Y | X_s)\) scores.

\(^4\)https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs