Metadata Shaping: Natural Language Annotations for the Tail

Simran Arora[1], Sen Wu[1], Enci Liu[1], and Christopher Ré[1]

[1]Stanford University
{simran,senwu,jesslec,chrismre}@cs.stanford.edu

October 16, 2021

Abstract

Language models (LMs) have made remarkable progress, but still struggle to generalize beyond the training data to rare linguistic patterns. Since rare entities and facts are prevalent in the queries users submit to popular applications such as search and personal assistant systems, improving the ability of LMs to reliably capture knowledge over rare entities is a pressing challenge studied in significant prior work. Noticing that existing approaches primarily modify the LM architecture or introduce auxiliary objectives to inject useful entity knowledge, we ask to what extent we could match the quality of these architectures using a base LM architecture, and only changing the data? We propose *metadata shaping*, a method in which readily available metadata, such as entity descriptions and categorical tags, are appended to examples based on information theoretic metrics. Intuitively, if metadata corresponding to popular entities overlap with metadata for rare entities, the LM may be able to better reason about the rare entities using patterns learned from similar popular entities. On standard entity-rich tasks (TACRED, FewRel, OpenEntity), with no changes to the LM whatsoever, metadata shaping exceeds the BERT-baseline by up to 5.3 F1 points, and achieves or competes with state-of-the-art results. We further show the improvements are up to 10x larger on examples containing tail versus popular entities.

1 Introduction

While recent language models (LMs) are remarkable at learning patterns seen frequently during training, they still suffer from performance degradation over rare *tail* patterns. In this work, we propose metadata shaping, a method to address the tail challenge by encoding properties shared between popular and tail examples *in the data itself.*

Tail degradation is particularly apparent in *entity-rich tasks*, since users care about very different entities and facts when using popular entity-centric applications such as search and personal assistants [4, 30, 33]. Given the importance of entity-rich applications, considerable effort has been devoted to methods for capturing entity knowledge more reliably. Many proposed approaches proceeding by changing the model architecture.

While early LMs relied heavily on feature engineering [21, 49], recent neural models could learn useful features during training [8], and shifted the focus to model engineering. As such, given data annotated with entity tags, recent methods for reasoning about entities typically enrich *base LMs* (e.g., Devlin et al. [12]) by modifying the architecture or introducing auxiliary objectives to better learn entity properties [25, 32, 37, 41, 46, 47, 51], or by combining multiple learned modules, each specialized to handle fine-grained reasoning patterns or subsets of the data distribution [6, 39].

These *knowledge-aware LMs* have led to impressive gains compared to base LMs on entity-rich tasks, however they raise a few challenges: additional pretraining is expensive, deploying and maintaining multiple specialized LMs can be memory and time-intensive, each LM needs to be optimized for efficient use (e.g., on-device), and the methods require training to adapt to changing facts or new entities. Further, these LMs learn about an entity from its individual occurrences during training, rather than explicitly encoding that patterns learned for one entity may be useful for reasoning about other *similar* entities. Implicitly learning subpopulations may be data-inefficient as the Wikidata knowledge base alone holds $\sim 100M$ entities and
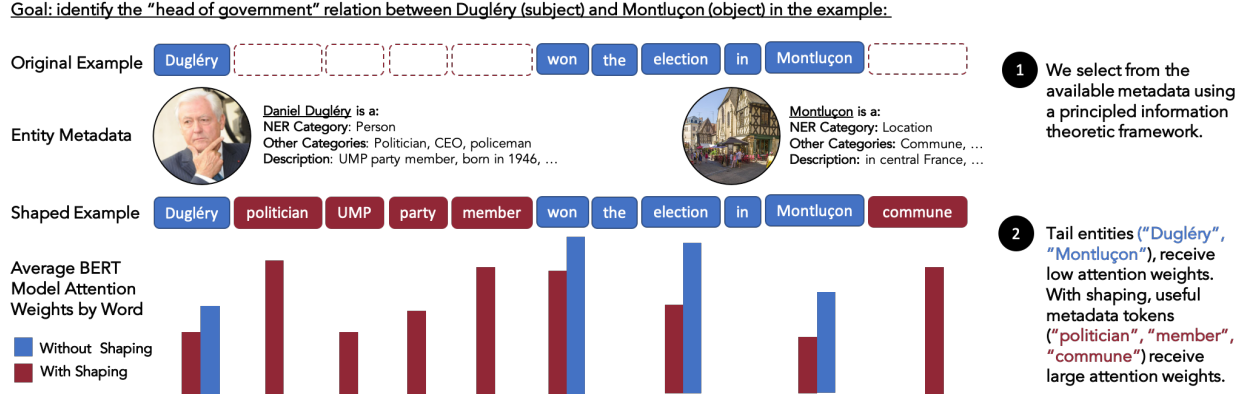
Figure 1: Metadata shaping inserts metadata (e.g., entity categories and descriptions) into the data at train and test time. The FewRel benchmark involves identifying the relation between a subject and object string. The above subject and object are unseen in the FewRel training data and the tuned base LM places low attention weights on those tokens. A base LM trained with shaped data has high attention weights for useful metadata tokens such as "politician". Weights are shown for words which are not stop-words, punctuation, or special-tokens.

unfortunately, 89% of the Wikidata entities do not even appear in Wikipedia, a popular source of unstructured training data for the LMs. [1]

Thus, we asked the question: to what extent could we match the quality of these architectures using a base LM architecture, and only changing the data? We propose some simple modifications to the data at train and test time, which we call metadata shaping, and find the method is surprisingly quite effective. Given unstructured text, there are several readily available tools for generating annotations at scale [16, 27], and knowledge bases contain entity metadata including categorical tags (e.g., Barack Obama is a "politician") and descriptions (e.g., Barack Obama "enjoys playing basketball"). Metadata shaping entails explicitly inserting retrieved metadata in the example as in Figure 1 and inputting the resulting *shaped example* to the LM. We find focusing on the data to be a compelling way to improve tail generalization. Our contributions are:

Simple and Effective Method We propose metadata shaping and demonstrate its effectiveness on entity-rich benchmarks. Metadata shaping, with simply an *off-the-shelf* base LM, exceeds the base LM trained on unshaped data by up to 5.3 F1 points and competes with state-of-the-art methods, which do modify the LM. Metadata shaping thus enables re-using well-studied and optimized base LMs (e.g., Sanh et al. [35]).

Tail Generalization We show that the performance improvements are concentrated on the tail — the observed improvement from shaping for examples containing tail entities is up to 10x larger than on the slice of examples containing popular entities across the evaluated tasks.[2] Metadata shaping simplifies the task to learning properties of *groups* of entities. Metadata establish "subpopulations", groups of entities sharing similar properties, in the entity distribution [9, 52]. For example on the FewRel benchmark [15], "Daniel Dugléry" (a French politician) appears 0 times, but "politician" entities in general appear > 700 times in the task training data. Intuitively, performance on a rare entity should improve if the LM has the explicit information that it is similar to an entity observed during training. This is data-efficient compared to learning properties of entities from individual occurrences.

Explainability Prior knowledge-aware LMs use metadata [2, 32], but do not explain when and why different metadata help. There is a rich set of tools for reasoning about data distributions, in contrast to neural

---

[1]Orr et al. [30] finds that a BERT based model needs to see an entity in on the order of 100 samples to achieve 60 F1 points when disambiguating the entity in Wikipedia text.

[2]Tail entities are those seen < 10 times during training, consistent with [13, 30].

architectures. Inspired by classic feature selection techniques [14], we conceptually explain the effect of metadata on generalization error.

We hope this work motivates further research on addressing the tail challenge through the data, alongside the model.

2 Method

This section introduces metadata shaping, including the set up and conceptual framework.

2.1 Objective

The goal of metadata shaping is to improve tail performance, using properties shared by popular and rare examples. For instance, this work investigates improving tail performance in entity-rich tasks using entity metadata (e.g., "politician") which are readily available for tail (e.g., "Daniel Dugléry") and popular *head* (e.g., "Barack Obama") entities. We define tail entities as those seen < 10 times during training, consistent with [13, 30].

Metadata are sourced by using off-the-shelf models (e.g., for entity linking, named entity recognition (NER), part-of-speech (POS)) [16, 27], heuristic rules, and metadata in knowledge bases (KBs) (e.g., Wikidata, Wordnet [29], and domain-specific KBs such as UMLS [5]) to tag unstructured text, making it is easy to annotate at scale. We consider an unrestricted set of metadata, rather than a predefined schema.

2.2 Set up

Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, and consider the labeled classification dataset $\boldsymbol{D} = \{(x_i, y_i)\}_{i=1}^n$ of n examples. Let $m \in \mathcal{M}$ denote a metadata token and let $\boldsymbol{M(x_i)}$ be the set of metadata tokens collected for example x_i. A shaping function $f_s : \mathcal{X} \to \mathcal{X}_s$ accepts an original example $x_i \in \mathcal{X}$ and produces a shaped example $s_i \in \mathcal{X}_s$ by inserting a subset of $\boldsymbol{M(x_i)}$ (See Figure 1). The downstream classification model $\hat{p}_\phi$ is learned from resulting shaped data to infer y_i from the shaped example s_i.

In this work, we use the following representative metadata shaping functions for all tasks. These functions range from applying signals to groups of examples to individual examples:

- **Structure tokens** Structure tokens indicate the spans of interest in the task. For example, inserting the special token $[ENTITY]$ surrounding the key entities in an entity-rich task. This is a common strategy, which can be viewed as an instance of metadata shaping.

- **Categorical tokens** Categorical tokens create subpopulations of entities (e.g., entities like *Dugléry* fall in the coarse grained *person* category, or finer grained category of *politician* entities). Off-the-shelf models (e.g., for NER and POS) provide coarse grained categories, and knowledge bases contain fine-grained categories (e.g., entity types and relations). Category tokens are consistent and frequent compared to words the original example.

- **Descriptive tokens** Descriptions give cues for rare entities and alternate views of popular entities. E.g., *Dugléry* is "a French politician and the son of a baker". In contrast to categories, descriptions are highly variable and likely to be unique for across entities.

2.3 Conceptual framework

Next we want to understand why inserting $m \in \boldsymbol{M(x_i)}$ for $x_i \in \boldsymbol{D}$ can improve tail performance. We measure the generalization error of the classification model $\hat{p}_\phi$ using the cross-entropy loss:

$$\mathcal{L}_{\text{cls}} = \mathbb{E}_{(x,y)}\big[-\log(\hat{p}_\phi(y|x))\big]. \tag{1}$$

Let $\Pr(y|x_i)$ be the true probability of class $y \in Y$ given x_i. Example x_i is composed of a set of linguistic patterns $\boldsymbol{K_i}$ (e.g., n-grams). We make the assumption that a pattern $k \in \boldsymbol{K_i}$ is a useful signal if it informs $\Pr(y|x_i)$. We thus parametrize the true distribution $\Pr(y|x_i)$ using the principle of maximum entropy [3]:

3

$$\Pr(y|x_i) = \frac{1}{Z(x_i)} \exp(\sum_{k \in \boldsymbol{K_i}} \lambda_k \Pr(y|k)). \tag{2}$$

where λ_k represents learned parameters weighing the contributions of patterns (or events) k and $Z(x_i)$ is a partition function ensuring $\Pr(y|x_i)$ represents a probability distribution. Therefore when evaluating $\hat{p}_\phi$, achieving zero cross-entropy loss between the true probability $\Pr(y|k)$ and the estimated probability $\hat{p}_\phi(y|k)$, for all k, implies zero generalization error overall.

Unseen Patterns Our insight is that for a pattern k that is unseen during training, which is common in entity-rich tasks,[3] the class and pattern are independent ($y \perp k$) under the model's predicted distribution $\hat{p}_\phi$, so $\hat{p}_\phi(y|k) = \hat{p}_\phi(y)$. With the assumption of a well-calibrated model and not considering priors from the base LM pretraining stage,[4] this probability is $\hat{p}_\phi(y) = \frac{1}{|\mathcal{Y}|}$ for $y \in \mathcal{Y}$.

Plugging in $\hat{p}_\phi(y) = \frac{1}{|\mathcal{Y}|}$, the cross-entropy loss between $\Pr(y|k)$ and $\hat{p}_\phi(y|k)$ is $\Pr(k) \log |Y|$. Our idea is to effectively replace k with another (or multiple) *shaped* pattern k', which has non-uniform $\hat{p}_\phi(y|k')$ and a lower cross-entropy loss with respect to $\Pr(y|k')$, as discussed next.

Inserting Metadata Consider the shaped example, $s_i = f_s(x_i)$, which contains new tokens from $\boldsymbol{M(x_i)}$, and thus contains a new set of linguistic patterns $\boldsymbol{K_i^s}$. Let $k_m \in \boldsymbol{K_i^s}$ be a pattern containing some $m \in \boldsymbol{M(x_i)}$. For the rare pattern (e.g., rare entity) k, if an associated pattern k_m (e.g., a metadata token for the rare entity) occurs non-uniformly across classes during training, then the cross-entropy loss between $\hat{p}_\phi(y|k_m)$ and $\Pr(y|k_m)$ is lower than the cross-entropy loss between $\hat{p}_\phi(y|k)$ and $\Pr(y|k)$. If k_m provides a high quality signal, shifting $\hat{p}_\phi(y|x_i)$ in the correct direction, performance of $\hat{p}_\phi$ should improve.

We can use the conditional entropy $\hat{H}(\mathcal{Y}|k)$ to measure the non-uniformity of k_m across classes. When k is unseen and $\hat{p}_\phi(y|k) = \hat{p}_\phi(y, k) = \hat{p}_\phi(y) = \frac{1}{|\mathcal{Y}|}$ (uniform), so the conditional entropy is maximized:

$$\hat{H}(\mathcal{Y}|k) = -\sum_{y \in \mathcal{Y}} \hat{p}_\phi(y, k) \log \hat{p}_\phi(y|k) = \log(|\mathcal{Y}|). \tag{3}$$

For non-uniform $\hat{p}_\phi(y|k_m)$, this conditional entropy decreases. Thus our application of using metadata for the tail connects back to classical feature selection methods [14], we seek the metadata providing the largest information gain. Next we discuss the practical considerations for selecting metadata.

Metadata Shaping Selection Entities are associated with large amounts of metadata $\boldsymbol{M(x_i)}$ — categories can range from coarse-grained (e.g., "person") to fine-grained (e.g., "politician" or "US president") and there are intuitively many ways to describe people, organizations, sports teams, and other entities. Since certain metadata may not be helpful for a task, and popular base LMs do not scale very well to long dependencies [31, 38], it is important to understand *which* metadata to use for shaping.

We want to select k_m with non-uniform $\hat{p}_\phi(y|k_m)$ across $y \in \mathcal{Y}$, or in other words lower $\hat{H}(\mathcal{Y}|k_m)$. By definition, the conditional probability $\Pr(y|k_m)$ is:

$$\Pr(y|k_m) = 2^{\mathrm{pmi}(y,k_m)} \Pr(y), \tag{4}$$

where we recall that the pointwise mutual information $\mathrm{pmi}(y, k_m)$ is defined as $\log\left(\frac{\Pr(y,k_m)}{\Pr(y)\Pr(k_m)}\right)$. The pmi compares the probability of observing y and k_m together (the joint probability) with the probabilities of observing y and k_m independently. Class-discriminative metadata reduce $\hat{H}(\mathcal{Y}|k)$.

Directly computing the resulting conditional probabilities after incorporating metadata in $\boldsymbol{D}$ is challenging since the computation requires considering all linguistic patterns across examples, generated by including m.

---

[3]For example, on the FewRel benchmark used in this work, 90.7%, 59.7% of test examples have a subject, object span which are unseen as the subject, object span during training.

[4]For this downstream analysis, we ignore occurrences in the pretraining corpus and learned similarities between unseen k and seen k'. Future work can incorporate those priors to refine the slice of unseen entities.

Algorithm 1 Metadata Token Selection

1: **Precompute Train Statistics**
2: **Input**: training data $\boldsymbol{D}_{train}$, metadata M
3: **for** each category $m \in M$ over $\boldsymbol{D}_{train}$ **do**
4:     Compute $\mathrm{pmi}(y, m)$ for $y \in \mathcal{Y}$.
5: **end for**
6: **for** each class $y \in \mathcal{Y}$ over D_{train} **do**
7:     Compute frequency f_y.
8: **end for**
9:
10: **Select Metadata for Sentence**
11: **Input**: x_i from $\boldsymbol{D}_{train}$ and $\boldsymbol{D}_{test}$, integer n.
12: Collect metadata $\boldsymbol{M(x_i)}$ for x_i.
13: **for** $m \in \boldsymbol{M(x_i)}$ **do**
14:     Compute $r_y = 2^{\mathrm{pmi}(m,y)} f_y$ for $y \in \mathcal{Y}$.
15:     Normalize r_y values to sum to 1.
16:     Compute entropy H_m over r_y for $y \in \mathcal{Y}$.
17: **end for**
18: Rank $m \in \boldsymbol{M(x_i)}$ by H_m.
19: **Return** the $\min(n, |\boldsymbol{M(x_i)}|)$ tokens with lowest H_m.

Instead we use simplistic proxies to estimate the information gain. In Algorithm 1, we focus on the subset of $\boldsymbol{K_i^s}$ containing individual metadata tokens m, and computes the entropy over $\hat{p}_\phi(y|m)$ for $y \in \mathcal{Y}$. Simple extensions to Algorithm 1, at the cost of additional computation, consider a broader set of k_m (e.g., n-grams containing m for $n > 1$), or *iteratively* select tokens by considering the correlations in the information gain between different metadata.

Metadata Shaping Noise Feature noising [40] is effective for structured prediction tasks to prevent overfitting. While regularization is typically applied directly to model parameters, Dao et al. [11], Xie et al. [45] regularize through the data. Since sources of metadata are not guaranteed to be accurate, and all entities may not be correctly linked or consistently categorized in the KB, we hypothesize that using metadata with diverse word choice and order (e.g., entity descriptions) and blank noising (e.g., by masking metadata tokens), can help reduce overfitting, and investigate this empirically in Section 4.

3   Experiments

In this section, we demonstrate that metadata shaping is general and effective, and in Section 4, we provide a detailed analysis of the framework. We apply the method to diverse entity-rich tasks and compare to current state-of-the art methods.

3.1   Datasets

We evaluate on standard entity-typing and relation extraction benchmarks used by baseline methods.

Entity Typing The entity typing task involves predicting the the applicable *types* for a given substring in the input example, from set of output types. We use the OpenEntity [7], which consists of a set of nine output types and is a multilabel classification task. We use a maximum of 25 metadata tokens for the main entity span first inserting category, then description tokens.

Relation Extraction Relation extraction involves predicting the relation between the two substrings in the input example, one representing a subject and the other an object. We use the FewRel and TACRED (Revisited) benchmarks for evaluation [2, 15, 50]. FewRel consists of a set of 80 output relations and TACRED

Benchmark	Train	Valid	Test
TACRED	68124	22631	15509
Category	54k/46k	16k/15k	9k/10k
Description	50k/43k	15k/14k	8k/9k
FewRel	8k	16k	16k
Category	8k/8k	16k/15k	16k/15k
Description	7k/8k	15k/16k	15k/16k
OpenEntity	1998	1998	1998
Category	674	674	647
Description	655	672	649

Table 1: We show the benchmark split sizes (row 1), and the # of examples tagged with category and description metadata (rows 2 and 3). We give numbers for the subject and object entity-span on relation extraction and the main entity-span for entity-typing. Tasks have different proportions of shaped examples.

includes 42 possible output relations. We use a maximum of 20 metadata tokens per subject, object entity, first inserting category, then description tokens.

For certain entity spans in these tasks, there are no linked entities (e.g., several are simply pronouns or dates). Table 1 gives task statistics. Section 4 considers different numbers and positions of inserted metadata tokens in the shaped examples.

3.2 Experimental Settings

Source of Metadata   We use a pretrained entity-linking model [30] to link the text in each task to an October 2020 dump of Wikidata. We use Wikidata and the first sentence of an entity's Wikipedia page to obtain descriptions. Additional details for the entity linking model and the metadata are in the appendix.

Model   We fine-tune a BERT-base model on metadata shaped data for each task, taking the pooled representation and using a linear prediction layer for classification [12]. We use cross-entropy loss for FewRel and TACRED and binary-cross-entropy loss for OpenEntity. All test scores are reported at the epoch with the best validation score and we use the scoring implementations released by [51]. Additional training details are in the appendix.

3.3 Baselines

The following knowledge-aware LMs are proposed by prior work for the evaluated tasks and currently state-of-the-art. **ERNIE**, [51] **LUKE** [47], **KEPLER** [41], **CokeBERT** [37], and **WKLM** [46] introduce auxilliary loss terms and require additional pretraining. Prior approaches also modify the architecture for example using alternate attention mechanisms (**KnowBERT** [32], **K-BERT**, [24], **LUKE**) or training additional transformer stacks to specialize in knowledge-based reasoning (**K-Adapter** [39]). **E-BERT** [33] does not require additional pretraining and uses entity embeddings which are aligned to the word embedding space. In Table 2, we compare to methods which use the same base LM, BERT-base, and external information resource, Wikipedia, for consistency.

3.4 End-to-End Benchmark Results

Main Results We simply use an *off-the-shelf* BERT-base LM [43], with no additional pretraining and fine-tuned on shaped data. Directly using the same three metadata shaping functions for each task, metadata shaping exceeds the BERT-base LM trained on unshaped data by 5.3 (FewRel), and 4.3 (TACRED), 3.0 (OpenEntity) F1 points. Metadata shaping is also competitive with SoTA baselines which *do* modify the LM and use the same Wikipedia resource (Table 2). For the baselines, we use reported numbers when available

Model	FewRel			TACRED			OpenEntity		
	P	R	F1	P	R	F1	P	R	F1
BERT-base	85.1	85.1	84.9	66.3	78.7	72.0	76.4	71.0	73.2
ERNIE	88.5	88.4	88.3	74.8	77.1	75.9	78.4	72.9	75.6
E-BERT$_{concat}$	88.5	88.5	88.5	-	-	-	-	-	-
K-BERT	83.1	85.9	84.3	-	-	-	76.7	71.5	74.0
KnowBERT$_{Wiki}$	89.2	89.2	89.2	**78.9**	**76.9**	**77.9**	78.6	71.6	75.0
CokeBERT	89.4	89.4	89.4	-	-	-	78.8	**73.3**	75.6
Ours (BERT-base)	**90.2**	**90.2**	**90.2**	75.9	75.6	76.3	**79.3**	**73.3**	**76.2**

Table 2: Test scores on relation extraction and entity-typing tasks. "Ours (Base LM)" is metadata shaping. All methods use the same base LM (BERT-base) and external information (Wikipedia) for consistency.

and Su et al. [37] reports two KnowBERT-Wiki results. We obtain other numbers using the released code (See Appendix).

The metadata, which are at different granularities, are more useful in combination. Ablating each shaping function for FewRel and OpenEntity, the task structure tokens, with no other shaping, provide a 26.3 (FewRel), 24.7 (OpenEntity) F1 point. boost. Category tokens for the entities tagged in the examples, and no descriptions, provides a 3.0 (FewRel), 2.5 (OpenEntity) F1 point boost. Descriptions of entities tagged in the examples, with no category tokens, provides a 1.6 (FewRel), 1.5 (OpenEntity) F1 point boost.

4 Analysis

Given the framework introduced in Section 2, here we study the following key questions for effectively using metadata shaping: **Framework (Section 4.1)** What are the reasons for different granularities of metadata? How should we select and combine metadata? **Tail Evaluation (Section 4.2)** What are the effects of metadata shaping on slices concerning tail vs. popular entities in the data? **Insertion Choices (Section 4.3)** How do different choices for inserting metadata into original examples affect performance?

4.1 Framework: Role of Metadata Types

Metadata Effects *Class-discriminative metadata correlates with reduced model uncertainty. With high quality metadata, as found in Wikidata, this results in improved classification performance.*

First to investigate the effects of metadata on model uncertainty, we compute the entropy of $\hat{p}_\phi$ softmax scores over the output classes as a measure of uncertainty, and compute the average across test set examples. We observe that lower uncertainty by this metric is correlated with improved classification performance (See Figure 2 (Top)).

We compute pointwise mutual information (pmi) scores for inserted metadata tokens as a measure of class-discriminativeness. We rank individual tokens k by $pmi(y, k)$ (for task classes y), computed over the training dataset. On FewRel, for test examples containing a top-20 pmi word for the gold class, the accuracy is 27.6% higher vs. the slice with no top-20 pmi words for the class. Notably, 74.1% more examples contain a top-20 pmi word for their class when pmi is computed on shaped data vs. when the training data is unshaped.

Metadata Token Selection *Simple information theoretic heuristics suffice to rank metadata by the information gain they provide, despite the complexity of the underlying contextual embeddings.*

We apply Algorithm 1 to select metadata tokens for the tasks, which ranks metadata tokens by their provided information gain. Given an entity with a set M of metadata tokens, our goal is to select n to use for shaping. We compare four selection approaches: using the highest ("High Rank") and lowest ("Low Rank") ranked tokens by Algorithm 1, random metadata from M ("Random"), and the most popular metadata tokens across the union of $M(x_i), \forall x_i \in D$ ("Popular"), selecting the same number of metadata tokens per example for each baseline. We observe that "High Rank" consistently gives the best performance, evaluated over

Benchmark	Strategy	Test F1
FewRel	BERT-base	84.9
	Random	87.3 ±0.8
	Popular	87.9 ±0.1
	Low Rank	87.8 ±0.4
	High Rank	**89.0** ±0.6
OpenEntity	BERT-base	73.2
	Random	74.3 ±0.7
	Popular	74.5 ±0.4
	Low Rank	74.1 ±0.4
	High Rank	**74.8** ±0.1
TACRED	BERT-base	72.0
	Random	73.8 ±1.6
	Popular	73.6 ±0.9
	Low Rank	73.3 ±1.0
	High Rank	**74.7** ±0.5

Table 3: Average and standard deviation over 3 random seeds. Each method selects up to n metadata tokens per entity. For FewRel, TACRED, $n = 3$ per subject, object. For OpenEntity $n = 2$ per main entity as 33% of OpenEntity train examples have ≥ 2 categories available (80.7% have ≥ 3 categories on FewRel). Note we use larger n for the main results (Table 2).

multiple seeds, and note that even "Random" yields decent performance vs. the BERT-baseline, indicating the simplicity of our method (Table 3).

Considering the distribution of selected category tokens under each scheme, the KL-divergence between the categories selected by "Low Rank" vs. "Popular" is 0.2 (FewRel), 4.6 (OpenEntity), while the KL-divergence between "High Rank" vs. "Popular" is 2.8 (FewRel), 2.4 (OpenEntity). Popular tokens are not simply the best candidates; Algorithm 1 selects discriminative metadata.

For OpenEntity, metadata are relatively sparse, so categories appear less frequently in general and it is reasonable that coarse grained types have more overlap with "High Rank". For e.g., "business" is in the top-10 most frequent types under "High Rank", while "non-profit" (occurs in 2 training examples) is in "Low Rank"'s top-10 most frequent types. Metadata tokens overall occur more frequently in FewRel (See Table 1), so finer grained types are also quite discriminative. The most frequent category under "Low Rank" is coarse grain, "occupation" (occurs in 2.4k training examples), but the top-10 categories under "High Rank" are finer grain, containing "director" and "politician" (each occurring in > 300 training examples).

Metadata Noise   *Noisier metadata can provide implicit regularization. Noise arises from varied word choice and order, as found in entity descriptions, or blank noising (i.e. random token deletion).*

Descriptions use varied word choice and order vs. category metadata.[5] To study whether shaping with description versus category tokens lead the model to rely more on metadata tokens, we consider two shaping schemes that us 10 metadata tokens: 10 category tokens and 5 category, 5 description. We observe both give the ~same score on FewRel, 89.8 F1 and 89.5 F1, and use models trained with these two schemes to evaluate on test data where 10% of metadata tokens per example are randomly removed. Performance drops by 1.4 F1 for the former and 1.0 F1 for the latter.

Blank noising [45] by randomly masking 10% of inserted metadata tokens during training leads to a consistent boost on OpenEntity: 0.1 ("High Rank"), 0.5 ("Popular"), 0.5 ("Low Rank") F1 points higher than the respective scores from Table 3 over the same 3 random seeds. We observe no consistent benefit from masking on FewRel. Future work could investigate advanced masking strategies, for example masking discriminative words in the training data.

---

[5]Over FewRel training data: on average a word in the set of descriptions appears 8 times vs. 18 times for words in the set of categories, and the description set contains 3.3x the number of unique words vs. set of categories.
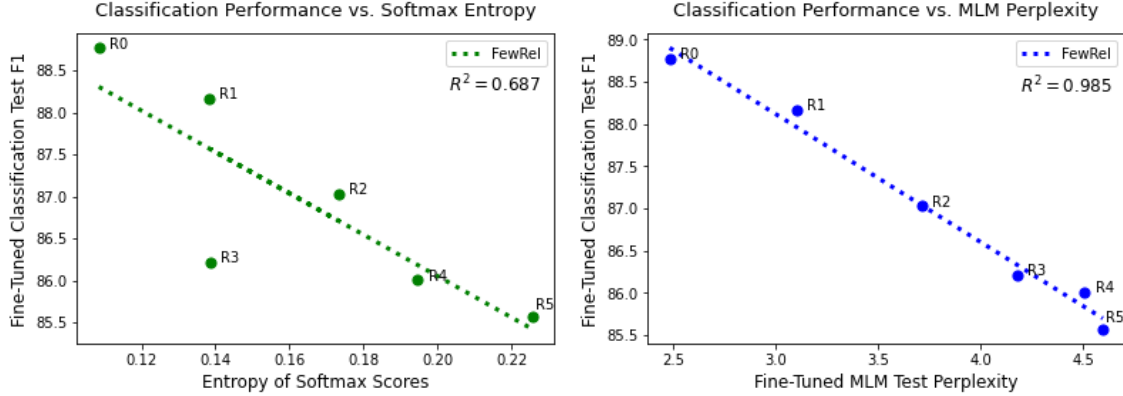
Figure 2: Test F1 for $\hat{p}_\phi$ (no additional pretraining) vs. average entropy of $\hat{p}_\phi$ softmax scores (Top) and vs. perplexity of an independent model $\hat{p}_\theta$ (w/ additional pretraining) (Bottom). $\hat{p}_\phi$ and $\hat{p}_\theta$ use the same shaped training data. Each point is a different metadata shaping scheme (median over 3 random seeds): for *R0* all inserted tokens are true category tokens associated with the entity. For *RX*, X true metadata tokens are replaced by randomly chosen tokens from the full vocabulary (i.e., noise, unlikely to be class-discriminative). For each point, the total number of metadata tokens (true plus random) is held constant per example.

Benchmark	R^2
FewRel	0.985
TACRED	0.782*
OpenEntity	0.956

Table 4: Correlation (R^2) between test F1 of $\hat{p}_\phi$ (no additional pretraining) vs. perplexity of $\hat{p}_\theta$ (w/ additional pretraining) for three tasks, using the procedure described in Figure 2. *Without one outlier corresponding to shaping with all random tokens ($R^2 = 0.02$ with this point).

Task Agnostic Metadata Effects   *Using metadata correlates with reduced task-specific model uncertainty, discussed in the prior analysis of the shaped model's softmax scores. Interestingly, metadata also correlates with reduced LM uncertainty in a task-agnostic way.*

Metadata shaping requires no additional pretraining for downstream tasks. However, in this section alone, we consider the effects of additional pretraining using shaped data on the base LM. Here, we perform additional masked language modeling (MLM) over the shaped task training data for the off-the-shelf BERT model to learn model $\hat{p}_\theta$. We minimize the following loss function and evaluate the model perplexity on the task test data:

$$\mathcal{L}_{\text{mlm}} = \mathbb{E}_{s \sim D, m \sim M, i \sim I} \big[-\log(\hat{p}_\theta(s_{m_i}|s_m/i)) \big]. \tag{5}$$

where I is the masked token distribution and s_{m_i} is the masked token at position i in the shaped sequence s_m.[6] Through minimizing the MLM loss, $\hat{p}_\theta$ learns direct dependencies between tokens in the data [48]. In Figure 2 (Bottom), Table 4, we observe a correlation between reduced perplexity for $\hat{p}_\theta$, and higher downstream performance for $\hat{p}_\phi$ across multiple tasks, both using the same training data. Lower perplexity indicates a higher likelihood of the data. It is interesting to observe this correlation since standard language modeling metrics such as perplexity differ from downstream metrics.

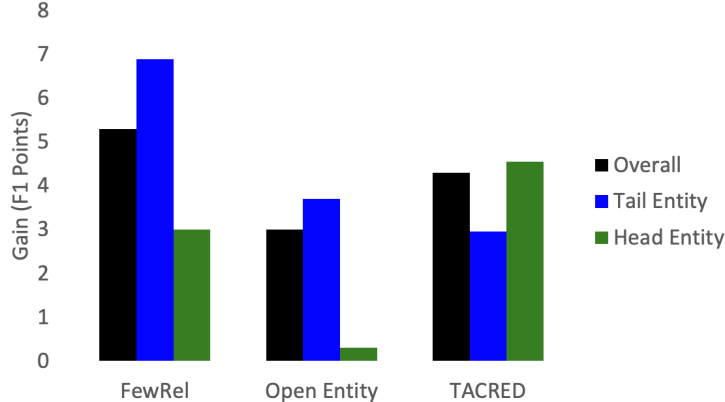[6] We use the Hugging Face implementation for masking and fine-tuning the BERT-base MLM [43].

Figure 3: The gain from the LM trained with metadata shaping vs. the BERT-base baseline split by the popularity of the entity span in the example.

4.2 Evaluation: Tail and Head Slices

Section 3 shows overall improvements from shaping. Here we consider fine-grained slices: the "tail" slice contains entity spans seen < 10 times and "head" slice contains those seen ≥ 10 times during training. We observe that much of the gain from shaping occurs on the tail test data (Figure 3).

For the unshaped BERT-base LM, the test F1 score is 93.3 (FewRel), 80.7 (OpenEntity) points on head examples, but just 82.5 (FewRel), 69.6 (OpenEntity) points on the tail. The shaped LM improves the tail F1 score by 6.9 (FewRel), 3.0 (OpenEntity) points and head F1 score by 3.7 (FewRel), 0.3 (OpenEntity) points. For TACRED, the base test F1 score is 68.7 on head examples and 73.7 on the tail. The shaped LM improves the tail F1 score by 3.0 and head F1 score by 4.6. A consideration for TACRED is that 42% of these head spans are stopwords (e.g., pronouns) or numbers; just 7% are for FewRel. [7]

Subpopulations *Metadata are helpful on the tail as they establish subpopulations.*

For the tail, we hypothesize that metadata establish subpopulations. Thus, if a pattern is learned for an entity-subpopulation occurring in the training data, the model may perform better on rare entities that also participate in the subpopulation, but were not individually observed during training. On FewRel, we take the top-20 TF-IDF words associated with each category signal during training as linguistic cues captured by the model for the category subpopulation. For example, "government" is in the top-20 TF-IDF words associated with the "politician" entity category. At test time, we select a slice of examples containing any of these words for any of the categories inserted in the example. Using metadata, the performance is 9.0, 3.5 F1 points higher on examples with unseen subject, object entities with vs. without a top-20 TF-IDF word for a subject, object category.

Metadata Effects on Popular Entities *For popular entities the LM can learn entity-specific patterns well, and be mislead by subpopulation-level patterns corresponding to metadata.*

Although we observe overall improvement on the popular entity slice, here we study the effect of metadata on popular entities within our conceptual framework. Let p be a popular pattern (i.e., an entity alias) in the training data, and let m be a metadata token associated with p. Intuitively, the LM can learn entity-specific patterns from occurrences of p, but coarser grained subpopulation-level patterns corresponding to m. If m and p are class-discriminative for different sets of classes, then m can mislead the LM.

To evaluate this, consider subject and object entity spans $p \in P$ seen ≥ 1 time during training. For test examples let $\mathcal{Y}_p$ be the set of classes y for which there is a $p \in P$ in the example with positive $\mathrm{pmi}(y, p)$, and define $\mathcal{Y}_m$ as the classes y for which there is a metadata token m with positive $\mathrm{pmi}(y, m)$ in the example. The set of examples where $\mathcal{Y}_p \neq \emptyset$, $\mathcal{Y}_m \neq \emptyset$, and $\mathcal{Y}_p$ contains the true class, but $\mathcal{Y}_m$ does not, represents the slice where metadata can mislead the model. On this slice of FewRel, the improvement from the shaped

---

[7]Based on unseen object spans for FewRel and TACRED, as $> 90\%$ subject spans are unseen.

model is 2.3 F1 points less than the improvement on the slice of all examples with $\mathcal{Y}_p \neq \emptyset$. $\mathcal{Y}_m \neq \emptyset$. When we narrow to p seen ≥ 10 times during training, the error rate from the shaped model is actually 1.1 F1 points *larger* than the unshaped model on the slice where $\mathcal{Y}_p$ contains the true class, but $\mathcal{Y}_m$ does not, supporting our intuition.

An example of entity-specific vs. subpopulation-level patterns in FewRel is: $p =$ "Thames River" is class-discriminative for $y =$ "located in or next to body of water", but its $m =$ "river" is class-discriminative for $y =$ "mouth of the watercourse".

4.3 Implementation Choices

This section studies the sensitivity of metadata shaping to the location in the example where metadata are inserted (e.g., the impact of position, long sequence lengths, and special tokens).

Token Insertion *We observe low sensitivity to increasing the context length and to token placement (i.e., inserting metadata directly-following the entity-span vs at the end of the sentence).*

We evaluate performance a the maximum number of inserted tokens per entity, n, increases. On FewRel, OpenEntity, performance tends to improve with n.[8] For FewRel, $n \in \{1, 5, 10, 20, 25\}$ gives $\{85.4, 86.4, 87.6, 88.5, 90.2\}$ test F1. On OpenEntity, $n \in \{1, 5, 10, 20, 25, 40\}$ gives $\{74.9, 75.7, 74.8, 74.5, 76.2, 75.8\}$ test F1. Shaped examples fit within the max sequence length and we observe low sensitivity to longer contexts.

The benefit of inserting metadata directly-following the entity span vs at the end of the example differed across tasks (e.g., for TACRED, placement at the end performs better, for the other tasks, placement directly-following performs better), though the observed difference is < 1 F1 point. We use the better-performing setting in Section 3 and for results on all tasks in Section 4, tokens are inserted directly-following the relevant entity span.

Boundary Tokens *Designating the boundary between original tokens in the example and inserted metadata tokens improves model performance.*

Inserting boundary tokens (e.g., "#") in the example, at the start and end of a span of inserted metadata, consistently provides a boost across the tasks. Comparing performance with metadata and boundary tokens to performance with metadata and no boundary tokens, we observe a 0.7 F1 (FewRel), 1.4 F1 (OpenEntity) boost in our main results. We use boundary tokens for all results in this work.

5 Discussion

We ultimately find that in certain cases, we can match the quality of knowledge-aware architectures using a base LM, and only changing the data. Benefits from metadata shaping include the ability to use simpler LMs, tackle the tail with better data efficiency by learning subpopulation-level (rather than only entity-specific) patterns, and provide a conceptual understanding by relying on the rich set of tools for reasoning about data. KnowBERT [32] considers NER, Wikpedia, and Wordnet [29], and Alt et al. [2] reports improvements from using NER and POS metadata, but neither explains the benefit or selection process.

However, there are also limitations to only changing the data. For instance, pretraining entity embeddings on a large number of examples allows the embedding to encode facts beyond the scope of the provided metadata tokens, and metadata shaping and the baselines (e.g., Zhang et al. [51]) rely on access to accurate sources of external knowledge such as the entity-linking model to obtain entity information. Finally, this work leverages bidirectional base LMs and focuses on classification — in future work we want to study how to best incorporate metadata with different types of LMs and tasks.

6 Related Work

Incorporating Knowledge in LMs As previously discussed, significant prior work incorporates knowledge into language modeling through changing the LM architecture or loss [24, 25, 32, 37, 39, 41, 46, 47, 51], in

---

[8]Per subject and object entity for FewRel, and per main entity for OpenEntity. I.e., $n = 10$ for FewRel yields a maximum of 20 total inserted tokens for the example.

contrast to our data focused approach.

Feature Selection   This work is inspired by techniques in feature selection based on information gain [14]. In contrast to traditional NLP feature schema [22, 28], metadata shaping annotations are expressed in natural language and the framework incorporates structured and unstructured annotations. We show classic feature selection methods based on information gain [3] are informative even when using complicated contextual embeddings. Information gain is a standard feature selection principle, and in the setting we study — the tail of entity-rich tasks — we contribute an application of the principle to explain how metadata can reduce generalization error.

Prompt-based Learning   Prompting typically involves changing the downstream task template to better elicit *implicit* knowledge from the base LM [23]. In contrast, the same metadata shaping functions apply directly to different task types and the method focuses on how to *explicitly* incorporate new, retrieved signals, not found in the original task. Prior work uses structured tokens in prompts for controllable text generation Aghajanyan et al. [1], Keskar et al. [20]; we consider a broader variety of structured data and provide intuition for selecting between structured data.

Data Augmentation   One approach to tackle the tail is to generate additional examples for tail entities [10, 42, 44]. However, this can be sample inefficient [17] since augmentations do not explicitly signal that different entities are in the same subpopulation, so the model would need view each entity individually in different contexts. In contrast, shaping does not generate new samples. Metadata shaping and prompting [36] may be viewed as implicit data augmentation.

Structured Data for Tail Generalization   Prior work leverages metadata for entities. Particularly relevant, Joshi et al. [19], Logeswaran et al. [26] demonstrate improvements on QA and named entity disambiguation (NED) respectively by using descriptions of entities, and Raiman and Raiman [34] uses entity types for entity-matching. Orr et al. [30] demonstrates that category metadata improves tail performance for NED, but requires custom architectures and storing trained embeddings for all categories and entities. We combine different granularities of metadata, provide a method that applies generally to downstream classification tasks, do not modify the LM, and provide conceptual grounding.

7   Conclusion

We propose metadata shaping to improve tail performance. The method is simple and general, yet competitive to model-based approaches for entity-rich tasks. We further show the empirical benefits are significant on the tail and formalize why metadata can reduce generalization error. While this work focused on entity-rich tasks, the method is not limited to this setting and we hope this work motivates further research on addressing the tail challenge through the data.

notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of NIH, ONR, or the U.S. Government.

References

[1] Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. Htlm: Hyper-text pre-training and prompting of language models. In *arXiv:2107.06955v1*, 2021.

[2] Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. Tacred revisited: A thorough evaluation of the tacred relation extraction task. In *ACL*, 2020.

[3] Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. In *Computational Linguistics*, 1996.

[4] Michael S Bernstein, Jaime Teevan, Susan Dumais, Daniel Liebling, and Eric Horvitz. Direct answers for search queries in the long tail. In *SIGCHI*, 2012.

[5] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. In *Nucleic Acids Research*, 2004.

[6] Vincent Chen, Sen Wu, Alex Ratner, J. Weng, and Christopher Ré. Slice-based learning: A programming model for residual learning in critical data slices. In *Advances in neural information processing systems*, pages 9392–9402, 2019.

[7] Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. Ultra-fine entity typing. In *arXiv:1807.04905v1*, 2018.

[8] Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. In *Journal of Machine Learning Research*, 2011.

[9] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019.

[10] Xiang Dai and Heike Adel. An analysis of simple data augmentation for named entity recognition. *Proceedings of the 28th International Conference on Computational Linguistics*, 2020.

[11] Tri Dao, Albert Gu, Alexander Ratner, Chris De Sa Virginia Smith, and Christopher Ré. A kernel theory of modern data augmentation. *Proceedings of the 36th International Conference on Machine Learning (PMLR)*, 2019.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[13] Karan Goel, Nazneen Rajani, Jesse Vig, Samson Tan, Jason Wu, Stephan Zheng, Caiming Xiong, Mohit Bansal, and Christopher Ré. Robustness gym: Unifying the nlp evaluation landscape. *arXiv preprint arXiv:2101.04840*, 2021.

[14] Isabelle Guyon and Andre Elisseeff. An introduction to variable and feature selection. In *Journal of Machine Learning Research*, 2003.

[15] Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *EMNLP*, 2018.

[16] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. Industrial-strength natural language processing in python. 2020.

[17] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. In *arXiv:1709.01450*, 2017.

[18] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*, 2019.

[19] Mandar Joshi, Kenton Lee, Yi Luan, and Kristina Toutanova. Contextualized representations using textual encyclopedic knowledge. In *ArXiv*, 2020.

[20] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv:1909.05858v2*, 2019.

[21] J. Lafferty, A. McCallum, , and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[22] Beth Levin. *English Verb Classes and Alternations*. The University of Chicago Press, 1993.

[23] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. In *arXiv:2107.13586*, 2021.

[24] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of AAAI*, 2020.

[25] Robert L. Logan, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *ACL 2019*, 2019.

[26] Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. Zero-shot entity linking by reading entity descriptions. In *arXiv:1906.07348v1*, 2019.

[27] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014.

[28] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. *Building a Large Annotated Corpus of English: The Penn Treebank*. 1993.

[29] George A Miller. Wordnet: a lexical database for english. In *Communications of the ACM*, 1995.

[30] Laurel Orr, Megan Leszczynski, Simran Arora, Sen Wu, Neel Guha, Xiao Ling, and Chris Ré. Bootleg: Chasing the tail with self-supervised named entity disambiguation. In *Arxiv*, 2020.

[31] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (PMLR)*, 2013.

[32] Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.

[33] Nina Poerner, Ulli Waltinger, and Hinrich Schutze. E-bert: Efficient-yet-effective entity embeddings for bert. *arXiv:1911.03681v2*, 2020.

[34] Jonathan Raiman and Olivier Raiman. Deeptype: multilingual entity linking by neural type system evolution. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[35] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv:1910.01108v4*, 2020.

[36] Teven Le Scao and Alexander M. Rush. How many data points is a prompt worth? In *NAACL HLT*, 2021.

[37] Yusheng Su, Xu Han, Zhengyan Zhang, Yankai Lin, Peng Li, Zhiyuan Liu, Jie Zhou, and Maosong Sun. Cokebert: Contextual knowledge selection and embedding towards enhanced pre-trained language models. In *arXiv:2009.13964*, 2021.

[38] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. In *arXiv:2011.04006v1*, 2020.

[39] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei1, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv:2002.01808v5 2020*, 2020.

[40] Sida I Wang, Mengqiu Wang, Stefan Wager, Percy Liang, and Christopher D Manning. Feature noising for log-linear structured prediction. *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[41] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *arXiv:1911.06136v3*, 2020.

[42] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *EMNLP 2019*, 2019.

[43] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 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, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[44] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation for consistency training. *NeurIPS 2020*, 2020.

[45] Zhang Xie, Sida I. Wang, Jiwei Li, Daniel Levy, Aiming Nie, Dan Jurafsky, and Andrew Y. Ng. Data noising as smoothing in neural network language models. *ICLR 2017*, 2017.

[46] Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. In *International Conference on Learning Representations (ICLR) 2020*, 2020.

[47] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. Luke: Deep contextualized entity representations with entity-aware self-attention. *EMNLP 2020*, 2020.

[48] Tianyi Zhang and Tatsunori Hashimoto. On the inductive bias of masked language modeling: From statistical to syntactic dependencies. *arXiv:2104.05694*, 2021.

[49] Yue Zhang and Joakim Nivre. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 2011.

[50] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. Position aware attention and supervised data improve slot filling. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[51] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*, 2019.

[52] Xiangxin Zhu, Dragomir Anguelov, and Deva Ramanan. Capturing long-tail distributions of object subcategories. In *CVPR*, 2014.

A Appendix

A.1 Dataset Details

Metadata   We tag original dataset examples with a pretrained entity-linking model from [30],[9] which was trained on an October 2020 Wikipedia dump with train, validation, test splits of 51M, 4.9M, and 4.9M sentences. FewRel includes entity annotations. The types we use as category metadata for all tasks are those appearing at least 100 times in Wikidata for entities their Wikipedia training data. This results in 23,413 total types. Descriptions are sourced from Wikidata descriptions and the first 50 words of the entity Wikipedia page.

Benchmarks   Benchmark datasets are linked here: https://github.com/thunlp/ERNIE.

A.2 Training Details

We use the pretrained BERT-base-uncased model for each task to encode the input text and a linear classification layer, using the pooled output representation. For TACRED, we also use a pretrained SpanBERT-large-cased model [18].

Entity Typing   Hyperparameters include seed 1, $2e-5$ learning rate, no regularization parameter and 256 maximum sequence length. For OpenEntity we use a batch size of 16 and use no gradient accumulation or warmup.

Relation Extraction   Hyperparameters include seed 42, $2e-5$ learning rate, no regularization parameter. For FewRel, we use For FewRel, we use batch size 16, 512 max sequence length, and no gradient accumulation or warmup. For TACRED, we use 256 maximum sequence length, batch size 48, and no gradient accumulation or warmup. We report the test score for the epoch with the best validation score for all tasks.

For Section 4 experiments, the random seeds for tasks are $\{42, 4242, 1010\}$ for all tasks.

A.3 Baseline Implementations

We produce numbers for key baselines which do not report for our benchmarks, using provided code.[10] [11]

- We produce numbers for KnowBERT-Wiki on TACRED-Revisited using a learning rate of $3e-5$, $\beta_2 = 0.98$, and choosing the best score for epochs $\in 1$, **2**, 3, 4 and the remaining provided configurations used for the original TACRED task.

- We produce numbers for ERNIE on TACRED-Revisited using the provided training script and configurations they use for the original TACRED task.

---

[9]https://github.com/HazyResearch/bootleg
[10]https://github.com/allenai/kb
[11]https://github.com/thunlp/ERNIE