CLESS: Contrastive Label Embedding Self-supervised Zero to Few-shot Learning from and for Small, Long-tailed Text Data

Nils Rethmeier [1 2] **Isabelle Augenstein** [2]

Abstract

For natural language processing 'text-to-text' tasks the prevailing approaches heavily rely on pretraining large self-supervised models on massive external data sources. This incurs exceptional pretraining data requirements and a diminished ability to pretrain over small datasets. However, challenges like minority information fairness and their machine learning relation to better long-tail, few and zero-shot concept prediction and evaluation remain open. We propose viewing pretraining from a miniaturisation perspective, able to *pretrain from multiple magnitudes smaller, 'task internal' data only*, while still strongly improving fully supervised, long-tail, few-shot and self-supervised zero-shot learning abilities. Our method, CLESS, uses contrastive label-embedding self-supervision to enable data efficient text encoder pretraining that is inherently zero-shot transferable and improves long-tail minority learning. Accordingly, we examine learning on a challenging long-tailed, low-resource, multi-label text classification dataset with noisy, highly sparse labels and many minority concepts. We find that long-tail, zero and few-shot learning markedly benefit from increasing 'dataset-internal' self-supervised training signal, which helps reduce reliance on large external sources.

1. Introduction

Despite the success of the self-supervised pretraining over *large-scale 'task-external' data* to learn NLP tasks in a 'text-to-text' framework (Raffel et al., 2020) there are two major, heavily interdependent, open challenges. One, a reliance on large to web-scale, 'end-task external pretraining data' (Liu et al., 2020; Yogatama et al., 2019) and extensive pretraining hardware resources (Hooker, 2020; Dodge et al., 2020) cause concerns about environmental costs (Strubell et al., 2019) and unintended contra-minority biases (Mitchell et al., 2020; Waseem et al., 2020)[1]. This has created a need for more data efficient methods (Schick & Schütze, 2020a; Şerbetci et al., 2020) with a shift in evaluation towards few to zero-shot probing (Brown et al., 2020; Talmor et al., 2019) or minority information coverage assessment (Mitchell et al., 2020; Buolamwini & Gebru, 2018). There are also increasing calls for more challenging "machine learning evaluation settings that are closer to real world (long-tail) problems", where most of the learnable *(tail) information is always* so *heavily limited* that *imbalanced, few or zero-shot learning become defaults*, as established by (Liu et al., 2019) and seen in Fig. 1 for the evaluation data setup we will use in this work. Crucially, "zero-shot probing evaluation measures information in an encoder, while fine-tuned probes introduce external biases into the evaluation results" as found by (Talmor et al., 2019; Elazar et al., 2020). Improved zero-shot and rare (few-shot) phenomenon prediction are paramount in disease detection, driving accident minimization and ethical bias considerations, but current *self-supervised* encoder pretraining methods require large data pretraining in NLP (Brown et al., 2020) or vision (Chen et al., 2020). Additionally, (Hooker et al., 2020a;b) found that the ability to encode long-tail (minority) information is a key ingredient in algorithmic contra-minority bias reduction because: (a) "minority fairness considerations coincide with long-tail information", while (b) "compressed models, loose long-tail,

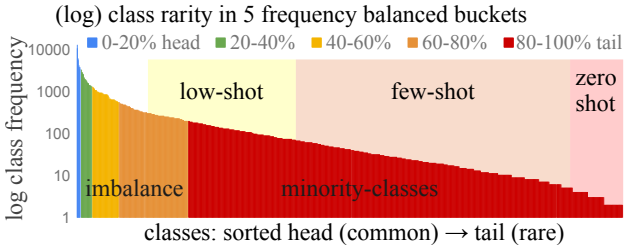

Figure 1. **Head to long-tail as 5 balanced class buckets:** We bucket classes by label frequency. Each bucket contains equally many label occurrences. Classes in a bucket are imbalanced and become few-shot or zero-shot towards the tail after train/dev/test splitting. Log class frequency, task data details in §4.

[1]German Research Center for AI, Germany [2]Copenhagen University, Denmark. Correspondence to: <nils.rethmeier@dfki.de>, <augenstein@di.ku.dk>. Preprint.

[1]Article on unpublished Gebru et al. paper – 2020 12 04

minority class prediction quality first", where (c) this effect is "often masked by traditional evaluation methodology". *Worse, these issues grow with data size, because the tail grows too – hence, simply adding more data and compute would only aggravate the problem.*

With CLESS, i.e. *Contrastive Label Embedding Self-Supervision*, we propose self-supervised pretraining over pseudo-label embeddings. CLESS is inherently capable of self-supervised zero-shot learning unlike (Zhang et al., 2018b; Pappas & Henderson, 2019; Jiang et al., 2019; Halder et al., 2020) which rely on supervised pretraining for zero-shot prediction or methods like SIMCLR (Chen et al., 2020) or (Kim et al., 2020) that can not actually zero-shot predict – details in §6.2. By predicting labels as word embeddings and words as pseudo-labels we can unify self-supervised and supervised learning as text input (word) and output (text label) prediction. This trains NLP like a 'text-to-text' task as in (Raffel et al., 2020) by learning to match 'text-embedding-to-label-embeddings', where positive and negative pseudo or real labels are sampled for contrastive training of a single binary match classifier. This classifier is reusable for any unseen (multi-)label tasks (infinite classes), where new labels can be described in words or embedded via (Bojanowski et al., 2017; Schick & Schütze, 2020b). Additionally, unlike (Chang et al., 2019; Halder et al., 2020; Bansal et al., 2020; Kim et al., 2020) does not require external web-scale pretraining data, and can be effectively pretrained from data set 3 orders of magnitude smaller than commonly used collections like Wikipedia – details in §6.3.

As a result, we observe several benefits due to the proposed self-supervised, contrastive pretraining. It markedly boost the prediction quality of minority long-tail classes. For few-shot learning we find large performance boosts from this pretraining and a that the model convergence twice as fast and without instability compared to training from scratch, especially in more extreme few-shot settings – §6.3. We also find that increased self-supervision in place of using more data, can increase zero-shot prediction (§6.2) and end-task long-tail prediction performance (6.1) over a robust baseline model that was optimized using standard generalization enhancing methods (Jiang et al., 2020).

2. Related Work

CLESS or Contrastive Label Embedding Self-supervision, calculatedly combines four major machine learning concepts – see §3 for details. (1) Label-embeddings to treat the prediction of inputs and outputs as well as self-supervision and supervision as the same task – i.e. labels are pretrained (word sequence embeddings) while the pretraining task prediction head is reused (transferred) for any new (self or supervised) task. (2) Proposing self-supervised label-embedding pretraining enables zero-shot prediction without ever seeing real labels. (3) We also combine supervised (CNN) and self-supervised (Transformer/ CNN) pretraining to enable efficient pretraining on small datasets and to enhance long-tail learning. (4) We optimize using noise contrastive estimation (Ma & Collins, 2018) to enable self-supervised CNN pretraining (or fine-tuning) and allow one to easily vary the amount of self-supervised learning signal – i.e. instead of adding more data, we can increase self-supervision learning signals. Finally, CLESS pretraining does not require special learning rate schedules, normalization layers, warm-up phases or modified optimizer as do BERT or RoBERTa (Devlin et al., 2019; Wang et al., 2020b). Batch sizes for CLESS models run on one 12GB GPU are between 128 and 1024 compared to 4 on BERT.

Label-embeddings (LE): Works by (Zhang et al., 2018b; Pappas & Henderson, 2019; Jiang et al., 2019) use *supervised label embedding CNN* pretraining to boost end-task performance and to allow zero-shot predictions. **Pretraining:** Works by (Zhang et al., 2018b; Pappas & Henderson, 2019; Jiang et al., 2019) use *solely supervised label embedding* pretraining. This is limited to *supervised* zero-shot transfer. (Kim et al., 2020; Chen et al., 2020) propose large-scale, task data external, self-supervised CNN pretraining *that outperforms BERT*, but is not zero-shot capable – i.e. require a new head per end-task. GPT3 (Brown et al., 2020), allows zero-shot prediction without labels, but *requires massive pretraining data*. Additionally, research by (Liu et al., 2020; Yogatama et al., 2019; Şerbetci et al., 2020) or when comparing (Wang et al., 2020a) with (Merity et al., 2017; Melis et al., 2020) suggests that Transformers struggle to learn from small data, where CNNs do not while being fast to train and iterate. **Zero-shot model requirements:** Zero-shot models like (Zhang et al., 2018b; Pappas & Henderson, 2019; Jiang et al., 2019) reuse the prediction head from self or supervised pretraining (SP). SP methods like SIMCLR (Chen et al., 2020) or (Kim et al., 2020) need to add a newly initialized prediction head per task and are thus not zero-shot capable. **Long-tail:** (Liu et al., 2017) use an label-embedding CNN for better many class prediction without pretraining. (Chang et al., 2019) combine two web-scale, externally pretrained models with label embeddings to further push many-class performance. However, they find that the BERT transformer token embeddings could not be used as labels embeddings, so they use ELMO word embeddings. Since the aforementioned CNN methods work with word embeddings and do not struggle with small datasets, this is the modelling we extend upon.

3. Model: unified self- and supervision via contrastive dense-to-dense text prediction

In this section, we propose to use label-embeddings, previously used for supervised learning only (Pappas & Hender-

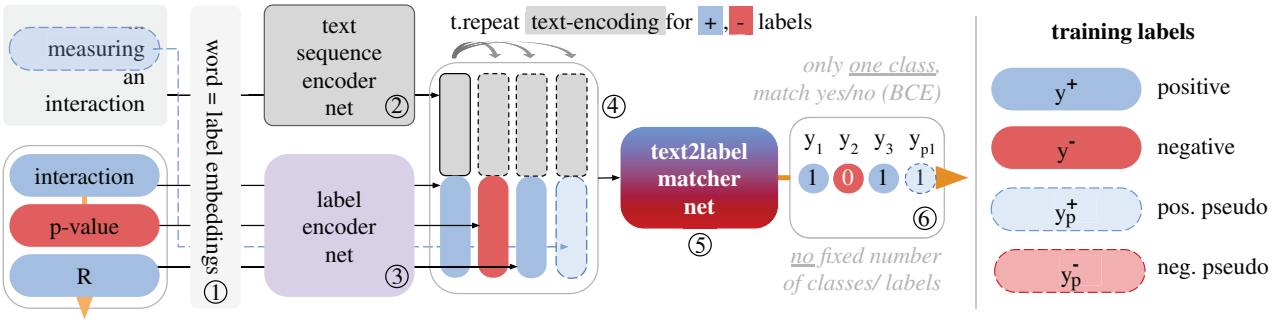

Figure 2. **Contrastive text-sequence-embedding-2-label-embedding matcher model:** A text ('measuring an interaction'), and positive ('interaction', R) or negative labels ('p-value') are encoded by the same word embedding layer E ①, where labels have word IDs for lookup. The text embeddings are then encoded by a sequence encoder T ②, while c labels are encoded by a label encoder L ③. Each text has multiple labels, so the text encoding $\boldsymbol{t}_i$ is repeated for, and concatenated with, each label encoding $\boldsymbol{l}^\circ_{i,l}$. The resulting batch of 'text-embedding, label-embedding' pairs $[[\boldsymbol{t}_i, \boldsymbol{l}^\circ_{i,1}], \ldots, [\boldsymbol{t}_i, \boldsymbol{l}^\circ_{i,c}]]$ ④ is fed into a 'matcher' classifier ⑤ that trains a binary cross entropy loss ⑥ on multiple (pseudo-)label (mis-)matches $\{0, 1\}$ for each text instance $\boldsymbol{t}_i$, resulting in a noise contrastive estimation objective (NCE). Words like 'measuring' provide self-supervised pseudo-labels (left). Positive and negative (pseudo-)labels are sampled from their own or other instances in a mini-batch. Unlike (Zhang et al., 2018a) we use a CNN for ②, negative sampling and self-supervision.

son, 2019; Zhang et al., 2018b), and exploit them for *self-supervised contrastive pretraining* on small-scale data. This enables contrastive self-supervised pretraining somewhat similar to methods used for large-scale models. However, we only use small-scale 'task-internal' data for pretraining, which requires orders of magnitude less data and compute than large-scale, 'task-external' pretraining approaches. Most NLP models translate back and forth between discrete words and continuous token embeddings, often using a softmax computation that is limited to predicting classes known at training time. To ease learning from small data, our **first core idea** *is that text input words $\boldsymbol{w}_i \in \boldsymbol{x}$ and labels $\boldsymbol{w}^\circ_{i,l}$ should be mapped into the same word representation space, i.e. drawn from a shared embedding look-up table E, to replace dense to sparse translations with embedding-to-embedding matching. Thus turns NLP from a discrete 'text-to-text' tasks, as proposed in (Raffel et al., 2020), into a 'dense(text)-to-dense(text)' task.* We thus replace learning instance labels y_i by their corpus-internally pretrained FastText or randomly initialised word embeddings $\boldsymbol{l}^\circ_i \in L$, while others (Pappas & Henderson, 2019) use text descriptions to form label embeddings as the vector average over description word embeddings. As a result, *pretraining word embeddings also pretrains (favourably initialising) label embeddings*. Unknown labels (words), in turn, can be inferred via methods like FastText subword embeddings (Bojanowski et al., 2017) or (Schick & Schütze, 2020b).

As outlined visually, left to right in Fig. 2, learning multi-label classification then becomes a contrastive learning problem of *matching the word-sequence embedding* $\boldsymbol{t}_i$ of text i ②, with its c label (word-sequence) embeddings $\boldsymbol{l}^\circ_i = \{\boldsymbol{l}^\circ_{i,1}, \ldots \boldsymbol{l}^\circ_{i,c}\}$ ③, by feeding c text-vs-label combinations $[[\boldsymbol{t}_i, \boldsymbol{l}^\circ_{i,1}], \ldots, [\boldsymbol{t}_i, \boldsymbol{l}^\circ_{i,c}]]$ ④ to a binary classifier M ⑤ for matching. This means that instead of predicting c classes at once, we predict a batch of c, single-class, binary classifications using binary cross entropy ⑥, where c needs not be constant across instances i. The details of steps ① to ⑥ are as follows. To train a binary classifier, we need both positive and negative labels. Thus, for each text instance $\boldsymbol{w}_i = \{w_a, \ldots w_z\}$ we want to classify, we need g positive labels $\boldsymbol{w}^-_i = \{w^+_1, \ldots w^+_g\} \in R^g$ and b negative labels $\boldsymbol{w}^+_i = \{w^-_1, \ldots w^-_b\} \in R^b$ to form a label selection vector $\boldsymbol{w}^\circ_i = \{\boldsymbol{w}^+ \oplus \boldsymbol{w}^-\} \in \mathbb{R}^{g+b}$. To indicate positive and negative labels, we also need a g sized vector of ones $\mathbf{1} \in \mathbb{R}^g$ and a b sized zero vector $\mathbf{0} \in \mathbb{R}^b$, to get a class indicator $\mathbb{I}_i = \{\mathbf{1} \oplus \mathbf{0}\} \in \mathbb{R}^{c=g+b}$. Both the text (word) indices $\boldsymbol{w}_i$ and the label indices $\boldsymbol{w}^\circ_i$ are passed through a shared 'word-or-label embedding' look-up-table E ①, after which they are passed through their respective encoder networks – T as text-sequence encoder, L as label encoder. Thus, the text-encoder produces a (single) text embedding vector $\boldsymbol{t}_i = T(E(\boldsymbol{w}_i))$ per text instance i ②. The label-encoder produces $c = g + n$ label embedding vectors ($\boldsymbol{l}^\circ_i$) that form a label-embedding matrix $\boldsymbol{L}_i = [\boldsymbol{l}^+_1, \ldots, \boldsymbol{l}^+_g, \boldsymbol{l}^-_1, \ldots, \boldsymbol{l}^-_b] \leftarrow L(E(\boldsymbol{w}^\circ_i))$ ③. As text-encoder T we use a (CNN→max-k-pooling→ReLU) sub-network, while the label-encoder L is simply an (average-pool) operation, since a single label ($\boldsymbol{w}^\circ_{i,j}$), e.g. 'multi'-'label', can consist of multiple words. To compare how similar the text-embedding $\boldsymbol{t}_i$ is to each label-embedding $\boldsymbol{l}^\circ_{i,j}$, we repeat $\boldsymbol{t}_i$ c times and combine text and label embeddings to get a text-vs-label-embedding matrix $\boldsymbol{M}_i = [[\boldsymbol{l}^+_{i,1}, \boldsymbol{t}_i], \ldots, [\boldsymbol{l}^-_{i,c}, \boldsymbol{t}_i]]$ ④ that is passed into the matcher network M ⑤ to produce a batch of c probabilities $\boldsymbol{p}_i = \{\sigma(M(\boldsymbol{M}_i)_1), \ldots, \sigma(M(\boldsymbol{M}_i)_c)\}$ ⑥. As the optimisation loss, we the use binary cross entropy (BCE) between $\boldsymbol{p}_i$ and $\mathbb{I}_i$, i.e. $\frac{1}{c}\sum_{l=1}^{c} \mathbb{I}_{i,l} \cdot log(\boldsymbol{p}_{i,l}) + (1 - \mathbb{I}_{i,l}) \cdot log(1 - \boldsymbol{p}_{i,l})$. Summing BCE over positive and negative (pseudo-)labels

results in noise contrastive estimation, as used in representation learning methods across fields (Ma & Collins, 2018).

Via pseudo-label embedding pretraining, a model can predict supervised labels absent prior supervision. This exploits both *transfer learning from inputs and labels*, using the match classifier as a learned similarity function. Positive labels $\boldsymbol{w}_i^+$ can be supervision labels. Negative labels $\boldsymbol{w}_i^-$ can be sampled from the positive labels of other instances $\boldsymbol{w}_j^+$ in the same batch, *which avoids needing to know the label set beforehand*. Since *labels are words*, we can sample positive words from the current and negative words from other text instances to get pseudo-labels. *Sampling pseudo-labels provides a straight-forward contrastive, partial autoencoding mechanism usable as self-supervision in pretraining or as zero-shot learner*. Because both real and pseudo labels are sampled words, the model does not need to distinguish between learning modes. Instead, learning is controlled by an out-of-model sampling routine for real supervision and pseudo self-supervision labels. This leads to a **second core idea:** *once inputs X and outputs Y are well initialised, the model Θ can also be better initialised by pretraining via self-supervision. As a result, we can learn supervised, few and zero-shot tasks in a unified manner*.

4. Small, long-tailed dense-to-dense text (label) prediction as a challenging test bed

Since it is our goal to research *better zero and few-shot learning approaches for small, long-tailed 'text-to-text' pretraining models*, we choose a small multi-label question tag prediction dataset as a test bed. We use the "Questions from Cross Validated"[2] dataset, where machine learning concepts are tagged per question. This dataset fulfills three of our requirements: it is small-scale, long-tailed, and entails solving a challenging, noisy 'text-to-text' prediction task. It is also fuzzily defined, as many real world problems are. In this case because tagging was crowd-sourced so important details like determining the correct amount of tags per question (label density) is hard even for humans. With early baseline multi-hot label models, we saw strong overfitting, despite using many standard generalization techniques as in (Jiang et al., 2020) – details in supplemental Tab. 1. Overfitting was reduced with supervised label embeddings, especially for setups with self-supervised label embedding pretraining, which essentially makes learning more divers and therefore harder to fit. This indicates that neural models struggle to find an easy shortcut to learning this task, further making the dataset an attractive test bed. *There is currently no published baseline for this task, and rather than reporting performance scalars, we will focus on analyzing learning curves, long-tail bucket analysis and learning modes*. The classes (tags) and input words are highly long-tailed (imbalanced). The first 20% of labels occur in only 7 'head' classes. Tags are highly sparse – at most 4 out of 1315 tags are labelled per question. Word embeddings are pretrained with FastText – details in appendix App. A.3. We use the labelled questions part of the dataset, which has 85k questions and 244k labels. What makes this problem particularly challenging is that 80% of the *least frequent labels* are distributed over 99.5% of classes, as an extreme long tail. The label density (% of active labels per question) is only 0.22% or $\approx 2.8/1305$ possible classes per instance. For a realistic evaluation setting, we split the dataset diachronically, using the 80% earliest documents for training, the next 10% for development, and the last 10% for testing.

Why not large external pretraining? Real-world, long-tailed datasets are always dominated by a low-learning-resource problem for most classes. This makes two things obvious: (A) that *model learning cannot simply be solved by using massive data sets as the long-tail problem grows as well*; (B) that *studying self-supervised pretraining on challenging,* but smaller*, long-tailed datasets such as this one, is useful for assessing an approach's ability to learn from complex, real-world data.* Massive data pretraining masks and thus prevents studying these effects. We thus evaluate the effects of self-supervision in a noisy low-resource setup, also as a response to recent critiques of the evaluation metrics used to assess Web-scale learning (Linzen, 2020; Yogatama et al., 2019). As (McCoy et al., 2019) shows, these evaluation setups are solvable by large-scale pattern overfitting, which, they find, leads to a 'Clever Hans effect', rather than real task progress.

5. Experimental setup and metrics

We analyse the *benefits of self-supervision for (a) long-tail, (b) few and (c) zero-shot learning in a noisy low-resource, long-tailed, multi-label classification setting*. In this section, we describe suitable evaluation metrics, then discuss results in the next section.

Long-tail evaluation metrics and challenges: Long-tail, multi-label classification is challenging to evaluate because (i) many measures like top-1 accuracy do not quantify performance losses on long-tailed minority classes as (Hooker et al., 2020c;b) point out. (ii) Measures like ROC_{AUC} largely overestimates performance under class imbalance(Davis & Goadrich, 2006; Fernández et al., 2018). (iii) Discrete measure like F-score do not scale to many classes, and require discretization threshold search under class imbalance. Luckily, the Average Precision score addresses these issues. It is defined as $AP = \sum_n (R_n - R_{n-1})P_n$, where P_n and R_n are the precision and recall at the nth threshold. Out of the different class weighting schemes for AP, we choose AP_{micro} and AP_{macro}, as they

[2]https://www.kaggle.com/stackoverflow/statsquestions

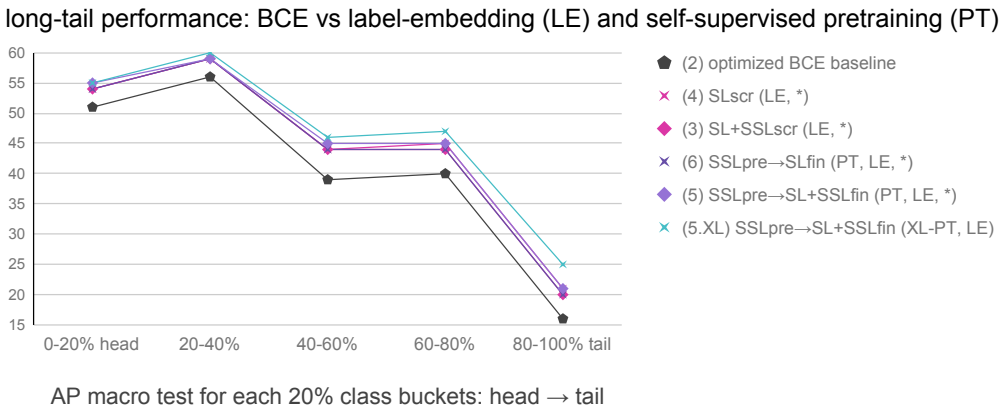

Figure 3. **Long-tail effects of base, label-embeddings and self-supervised pretraining (XL):** When reporting AP_macro of the five 20% head to tail class bins, the BCE multi-label objective performs worst. Label-embedding NCE (LE, (4-6)) markedly boosts performance, especially on the long-tail. When using label embeddings for self-supervised pretraining with the same network parameters (*) for all LE models, there is no boost. However, when pretraining with more parameters ((5.XL) – see `larger net, 3.3x pseudo-labels` in the zero-shot learning Fig. 5), we see substantial long-tail performance boosts (turquoise, upper most curve).

are the most pessimistic (hardest to increase) measures to reduce optimistic evaluation. We also report the macro-averaged Brier-Score (BS) over all classes, as a scalable, compute-efficient measure of classifier calibration. While more accurate measures exist, computing them is involved and they are meant to to optimise for a specific supervised dataset, which is not our goal. For both measures, we use their popular scikit-learn implementations[3].

6. Results

In this section, we first analyse how supervised long-tail performance can positively be affected by supervised label embeddings (real labels) and self-supervised label embeddings (pseudo labels) compared to a standard multi-hot prediction baseline model. Then, we demonstrate how the amount of self-supervision pretraining signal and model size affect zero-shot learning. Finally, we analyse the benefits of 'dataset-internal', self-supervised pretraining for few-shot learning. *Test score curves are reported for models that have the best dev set average precision score AP_{micro} over all classes.* Before running experiments, we wanted to establish guessing performance by computing a ZeroR classifier. ZeroR simply predicts the majority label per class and is a common baseline for assessing performance under class imbalance. The ZeroR AP_{micro} and AP_{macro} on this dataset are 0.2%, since out of the 1315 classes, maximally four classes are active per instance. In other words, 'random' guessing performance is zero, already indicating that the task may be difficult to learn.

[3] `https://scikit-learn.org/stable/modules/model_evaluation.html`

6.1. Label-embedding (pretraining) boosts the long-tail

In this section we analyse the effects of using supervised label-embeddings and self-supervised pretraining with words as pseudo-labels. Plotting the average precision of 1305 would be unreadable. Instead, we sort classes from frequent to rare and assign them to one of five 20% class frequency buckets, such that each bucket has the same amount of positive labels (label occurrences) in it. As seen in the intro Fig. 1, this means that the head $0-20\%$ bucket (left, blue) has very few, frequent classes, while tail buckets $20-40\% \ldots 80-100\%$ have increasingly more classes (right, orange, red) that contain increasingly more minority classes. We bucket classes to balance label frequency between buckets, thus making buckets directly comparable.

Label-embeddings increases overall head to tail performance: In Fig. 3 we see how different models perform over the class head-to-tail buckets defined in Fig. 1. To find model parameter amounts we increased model width for the base model (2) until we hit diminishing dev set AP_{micro} performance returns – this model can only be trained from scratch. Starting from these baseline hyperparameters in (2), we then further increased model width for label embedding (LE), 'from scratch models trained' models (3,4) until we again hit diminishing returns. This means that the label embedding (LE) models could use the increase in filters effectively, while the baseline could not. Models (5,6) first pretrain using self-supervision over pseudo label embeddings and are then fine-tuned using supervised label embeddings of real task labels. Label embedding models (3,4,5,6) all use the same hyperparameters to make sure performance differences only originate from comparing

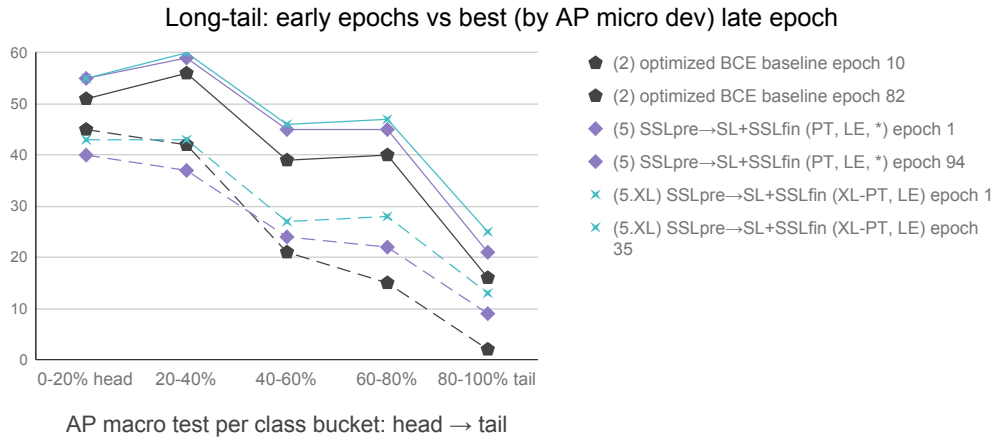

Figure 4. **The long-tail is learned during later epochs:** AP_{macro} performance over five frequency balanced 20% head to tail class buckets. All methods (2-5.XL) learn the head classes ($0-20\%$) first – during early epochs. Self-supervised pretraining (5, 5.XL), via pseudo label-embedding NCE (LE, PT), learn the long-tail (see $60-100\%$) even in the first epoch (- - dashed line) of supervised learning. The baseline (2) struggles to learn this long-tail at epoch 10 and until its epoch 82 – i.e. its optimal dev set score epoch.

self-supervised pretraining to training from scratch – i.e. to ensure a controlled experiment.

The worst model (black line) is the baseline CNN multi-hot, BCE loss model (2), that we optimized using common generalization tricks as described in (Jiang et al., 2020). Performance improves for all buckets when training with from scratch with label embeddings (LE) (3,4). However, training from scratch adding pseudo-label 'from scratch training' does not increase performance. In (5,6) we see that self-supervised pretraining (pre) with pseudo label followed by supervised label embedding fine-tuning tuning only slightly boosts long-tail performance.

Pretraining with more self-supervision signal boosts minority (tail) learning: We suspected, that since pretraining uses input vocabulary words as pseudo labels, it also needs more parameter capacity to store the increase learning signal diversity. Hence, in model (5.XL) we increased both the model capacity (width) and the amount of pretraining of sampled pseudo label learning signals. Firstly, this increase in model width and label amount resulted in substantially increased zero-shot performance after self-supervised pretraining, as we explain in detail in §6.2. Secondly, this increase in self-supervised pretraining diversity and model size also markedly increases long-tail performance of minority classes (60-100% buckets) after fully supervised fine-tuning via real label embeddings, seen as model (5.XL) in Fig. 3. This demonstrates that improved self-supervised zero-shot performance translates into better supervised end-task fine-tuning performance. Thus, the benefits of model size increase are limited by the amount and self-supervised training signal diversity, as we expected. However, it also tells us that *a simple increase in label embedding self-supervision can improve minority prediction performance*. This helps in decreasing the long-tail, minority information erasure problems posed in (Hooker et al., 2020b;c;a), and in reducing the need for large scale, external pretraining data collections. Lastly, it may also explain why popular large-scale pretraining and models perform so well rather than simply assuming them to be overparameterized. Finally, the overall-buckets performance of the (5.XL) model is only .8 AP_micro percent points better than the (5) model. This demonstrates that summary metrics, without long-tail analysis, can easily hide important minority bias learning problems as also pointed out by (Hooker et al., 2020c).

The head is learned first (in early epochs), pretraining learns the tail much faster: In Fig. 4 we compare early epoch training with late (optimal) epoch test scores per class bucket. We see that all models learn the head classes during the first epochs (- - dashed line). Methods (5, 5.XL) that use label-embedding (LE) and self-supervised pretraining (PT), start learning the long-tail during the first epoch, while BCE multi-label baseline (2) does not start learning the long-tail until epoch 10, and even then at a much lower performance than the pretrained label-embedding methods. Finally, we see that pretraining a larger model with more self-supervised pretraining signal (5.XL or "larger net, 3.3x labels" in Fig. 5) increases performance on the long-tail, even during the first epochs.

Self-supervised label-embedding pretraining boosts long-tail performance: We thus conclude, that self-supervised pretraining helps us learn the long-tail better, and faster – i.e. even after a single epoch of supervised training. This is a useful property in supervised learning scenarios where data or computation cycles are limited.

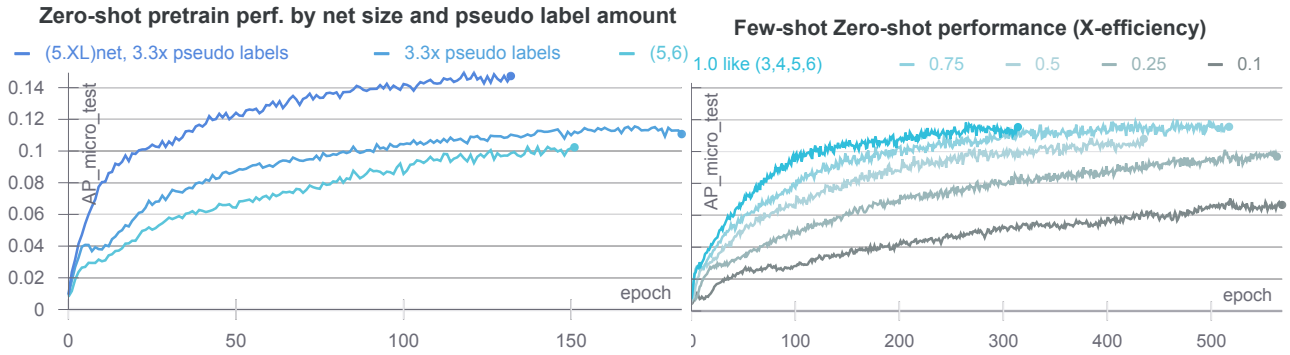

Figure 5. **Zero-shot performance by model size, signal amount and pretraining data amount (X-efficiency): Left plot:** When using the same label and parameter amount as for the 'joint self+supervised train from scratch' models (3,4), allowing more self-supervision pseudo labels (left middle curve) and widening the network (left top curve) noticeably boosts zero-shot performance (supervised $AP_{micro\_dev\ and\ test}$). **Right:** when using less pretraining data text (few-shot on inputs X), zero-shot works, but *takes longer*.

6.2. Zero-shot: more self-supervision is better and for 'low-resource' zero-shot pretrain longer

In this experiment, we study how the number of self-supervised labels (signal) and the model width used for self-supervised pretraining affects zero-shot performance on the end-task test set. In Fig. 6, we see that when using the shared hyperparameter configuration for label embedding methods (same for 3,4,5,6), pretraining gets the lowest zero-shot performance. When increasing the number of self-supervised word pseudo-labels from 150 to 500 (3.3x), the model performs better (middle curve), while not using more parameters – so *increasing self-supervision signals is beneficial*. When additionally tripling the network's sequence and label encoder width and doubling the label match classifier size, zero-shot performance increases even more (top curve). This indicates that *for zero-shot learning performance from pretraining, both the amount of training signals and model size have a significant impact*. While increased model size has been linked to increased zero-shot performance of Web-scale pretrained models like GPT3 (Brown et al., 2020), the influence of signal amount on zero-shot learning is much less well understood, because large-scale pretraining research often increases training data size when changing self-supervision, as outlined by (Liu et al., 2020).

Finally, in Fig. 5 we see that when pretraining our model for zero-shot prediction on only portions (100%, 75%, .50%, 25% and 10%) of the training text inputs X, i.e. an increasingly low-resource zero-shot setting, we still converge towards comparable full zero-shot performance (if we had not stopped early). However, each reduction in training size multiplies the required training time – when using the same number of self-labels. Thus, to produce the right side plots, we also allowed for more waiting epochs in early stopping. *This provides a promising insight into self-supervised pretraining on small datasets, which, if designed appropriately, can be used to pretrain well-initialised models for supervised fine-tuning and few-shot learning from very small text sizes.*

6.3. Few-shot: pretrain for better long-tail, low-resource, few-shot learning

In this section, we present evidence that even in a data-limited, long-tailed setting, self-supervised *'data-internal'* pretraining: (a) increases few-shot learning performance of subsequent fine-tuning, while (b) improving learning speed and stability. This demonstrates that small data pretraining has similar benefits as large-scale pretraining (Brown et al., 2020; Schick & Schütze, 2020a). In Fig. 6, when using the share models label embedding hyperparameters (same hyperparameters for 3,4,5,6), we now compare training from scratch (4) as before (pretraining off, left), with pretraining via self-supervised word pseudo-labels, and then fine-tuning on the supervised training labels (6) of the end-task (pretraining on). Note that our model architecture (Fig. 2) does not distinguish between self-supervised and supervised labels, which means that during self-supervised pretraining, we sample as many word pseudo-labels as real labels during supervised fine-tuning (or when supervising from scratch).

When fine-tuning the pretrained model on an increasingly difficult Few-Shot portion of (100%), 75%, 50%, 25% and only 10% of the supervised training data, we see large AP_{micro_test} performance improvements compared to training from scratch in Fig. 6. On the right, in Fig. 6, we see that the pretrained models start with a higher epoch-0 performance, train faster, are more stable and achieve a markedly better few-shot end performance than the left-hand 'from scratch' setting. This provides evidence to answer the underlying question: *"Do we really need more data for pretraining or can we simply increase self-supervision?"*. Very recent work by (Bansal et al., 2020) has investigated this question for large-scale, self-supervised pretraining, where

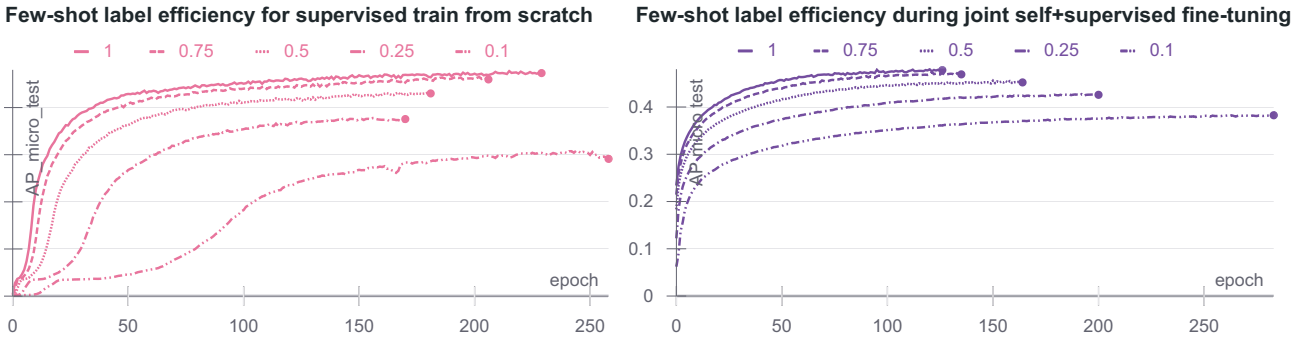

Figure 6. **Few-shot learning: Best training from scratch (left) vs. best fine-tuned (right)**: AP_{micro_test} curves for different few-shot portions: 100%, 75%, 50%, 25%, and 10% of training samples. *'Dataset-internal' pretraining via self-supervision (right) markedly improves few-shot learning performance, speed and stability* compared to training from scratch (left).

they showed that increasing self-supervision to create "a richer learning signal" benefits few-shot performance of large models. *Our results demonstrate that this is also the case for small-scale, non-Transformer pretrained models, even under a much more challenging long-tailed learning setting* than (Bansal et al., 2020) examined. However, to better understand the benefits of using more self-supervised training signals and its relation to model size, we examine the zero-shot performance of our pretraining approach in regards to *label (signal) amount, network width and zero-shot X data-efficiency* (low-resource zero-shot performance) – i.e. zero-shot performance when pretraining on fractions of inputs X to forcibly limit self-supervision.

7. Conclusion

We showed that label-embedding prediction, modified for self-supervised pretraining on a challenging long-tail, low-resource dataset substantially improves low-resource few and zero-shot performance. We find that increased self-supervision, in place of increased data size or resorting to large-scale pretraining, strongly boosts few and zero-shot performance, even in challenging settings. In future, we envision that the proposed methods could be applied in scenarios where little in-domain (pre-)training data is available, e.g. in medicine (Şerbetci et al., 2020), and where new labels rapidly emerge at test time, e.g. for hashtag prediction (Ma et al., 2014). The code and data splits will be published on `https://github.com`.

References

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. Self-supervised meta-learning for few-shot natural language classification tasks. In *EMNLP*, pp. 522–534. ACL, 2020. doi: 10.18653/v1/2020.emnlp-main.38. URL `https://www.aclweb.org/anthology/2020.emnlp-main.38`.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *TACL*, 2017. URL `https://transacl.org/ojs/index.php/tacl/article/view/999`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL `https://arxiv.org/abs/2005.14165`.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson (eds.), *PMLR*, volume 81 of *Proceedings of Machine Learning Research*, pp. 77–91, New York, NY, USA, 23–24 Feb 2018. URL `http://proceedings.mlr.press/v81/buolamwini18a.html`.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit Dhillon. X-bert: extreme multi-label text classification with using bidirectional encoder representations from transformers. *NeurIPS*, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. URL `https://arxiv.org/abs/2002.05709`.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In *Machine*

Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006, pp. 233–240, 2006. doi: 10.1145/1143844.1143874. URL https://doi.org/10.1145/1143844.1143874.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of NAACL-HLT*. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah A. Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *CoRR*, abs/2002.06305, 2020. URL https://arxiv.org/abs/2002.06305.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals, 2020.

Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from Imbalanced Data Sets*. Springer, 2018. ISBN 978-3-319-98073-7. doi: 10.1007/978-3-319-98074-4. URL https://doi.org/10.1007/978-3-319-98074-4.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 3202–3213, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. URL https://www.aclweb.org/anthology/2020.coling-main.285.

Sara Hooker. The hardware lottery, 2020. URL https://arxiv.org/abs/2009.06489.

Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. What do compressed deep neural networks forget?, 2020a.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. Characterising bias in compressed models, 2020b.

Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. Characterizing and mitigating bias in compact models. In *WHI Workshop*. ICML, 2020c. URL http://whi2020.online/static/pdfs/paper_73.pdf.

Huajie Jiang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Transferable contrastive network for generalized zero-shot learning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 9764–9773, 2019. doi: 10.1109/ICCV.2019.00986. URL https://doi.org/10.1109/ICCV.2019.00986.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020. URL https://openreview.net/forum?id=SJgIPJBFvH.

Kang-Min Kim, Bumsu Hyeon, Yeachan Kim, Jun-Hyung Park, and SangKeun Lee. Multi-pretraining for large-scale text classification. In *Findings of EMNLP*, pp. 2041–2050. ACL, 2020. doi: 10.18653/v1/2020.findings-emnlp.185. URL https://www.aclweb.org/anthology/2020.findings-emnlp.185.

Tal Linzen. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of ACL*, 2020. URL https://www.aclweb.org/anthology/2020.acl-main.465/.

Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pp. 115–124, 2017. doi: 10.1145/3077136.3080834. URL https://doi.org/10.1145/3077136.3080834.

Qi Liu, Matt J. Kusner, and Phil Blunsom. A survey on contextual embeddings. *CoRR*, abs/2003.07278, 2020. URL https://arxiv.org/abs/2003.07278.

Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *IEEE CVPR*, pp. 2537–2546, 2019. doi: 10.1109/CVPR.2019.00264. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Liu_Large-Scale_Long-Tailed_Recognition_in_an_Open_World_CVPR_2019_paper.html.

Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 3698–3707, 2018. doi:

10.18653/v1/d18-1405. URL https://doi.org/10.18653/v1/d18-1405.

Zongyang Ma, Aixin Sun, Quan Yuan, and Gao Cong. Tagging Your Tweets: A Probabilistic Modeling of Hashtag Annotation in Twitter. In Jianzhong Li, Xiaoyang Sean Wang, Minos N. Garofalakis, Ian Soboroff, Torsten Suel, and Min Wang (eds.), *CIKM*, pp. 999–1008. ACM, 2014. ISBN 978-1-4503-2598-1. URL http://dblp.uni-trier.de/db/conf/cikm/cikm2014.html#MaSYC14.

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://www.aclweb.org/anthology/P19-1334.

Gábor Melis, Tomás Kociský, and Phil Blunsom. Mogrifier LSTM. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020. URL https://openreview.net/forum?id=SJe5P6EYvS.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Margaret Mitchell, Dylan Baker, Nyalleng Moorosi, Emily Denton, Ben Hutchinson, Alex Hanna, Timnit Gebru, and Jamie Morgenstern. Diversity and inclusion metrics in subset selection. In *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, pp. 117–123, 2020. doi: 10.1145/3375627.3375832. URL https://doi.org/10.1145/3375627.3375832.

Nikolaos Pappas and James Henderson. GILE: A generalized input-label embedding for text classification. *Trans. Assoc. Comput. Linguistics*, 7:139–155, 2019. URL https://transacl.org/ojs/index.php/tacl/article/view/1550.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Timo Schick and Hinrich Schütze. It's not just size that matters: Small language models are also few-shot learners. *CoRR*, abs/2009.07118, 2020a. URL https://arxiv.org/abs/2009.07118.

Timo Schick and Hinrich Schütze. Rare words: A major problem for contextualized embeddings and how to fix it by attentive mimicking. In *Proceedings of AAAI*. AAAI Press, 2020b. URL https://aaai.org/ojs/index.php/AAAI/article/view/6403.

Oğuz Necip Şerbetci, Sebastian Möller, Roland Roller, and Nils Rethmeier. Efficare: Better prognostic models via resource-efficient health embeddings. In *AMIA Annual Symposium*. PubMed, 2020. URL https://www.medrxiv.org/content/early/2020/07/26/2020.07.21.20157610.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3645–3650, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1355. URL https://www.aclweb.org/anthology/P19-1355.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. olmpics - on what language model pre-training captures. *CoRR*, abs/1912.13283, 2019. URL http://arxiv.org/abs/1912.13283.

Chenguang Wang, Zihao Ye, Aston Zhang, Zheng Zhang, and Alexander J. Smola. Transformer on a diet. *CoRR*, abs/2002.06170, 2020a. URL https://arxiv.org/abs/2002.06170.

Sinong Wang, Madian Khabsa, and Hao Ma. To pretrain or not to pretrain: Examining the benefits of pretraining on resource rich tasks, 2020b.

Zeerak Waseem, Smarika Lulz, Joachim Bingel, and Isabelle Augenstein. Disembodied machine learning: On the illusion of objectivity in nlp. anonymous preprint under review, 2020.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. Learning and evaluating general linguistic intelligence. *CoRR*, 2019. URL http://arxiv.org/abs/1901.11373.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. Multi-task label embedding for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*,

pp. 4545–4553, 2018a. URL https://www.aclweb.org/anthology/D18-1484/.

Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. Multi-task label embedding for text classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4545–4553, Brussels, Belgium, October-November 2018b. Association for Computational Linguistics. doi: 10.18653/v1/D18-1484. URL https://www.aclweb.org/anthology/D18-1484.