2/9/2026 1:09:09 PM

Compare Results

Old File:		New File:
2024.emnlp-main.1055.pdf	versus	**2024_emnlp-main_1055.pdf**
16 pages (14.23 MB)		**18 pages (387 KB)**
		2/8/2026 5:29:40 AM

Total Changes

41

Content

9 Replacements

17 Insertions

15 Deletions

Styling and Annotations

0 Styling

0 Annotations

Go to First Change (page 1)

Getting The Most Out of Your Training Data: Exploring Unsupervised Tasks for Morphological Inflection

Abhishek Purushothaman[1] Adam Wiemerslage[2] Katharina von der Wense[2,3]
[1]Georgetown University [2]University of Colorado Boulder [3]Johannes Gutenberg Universit
abhishek@cs.georgetown.edu

February 8, 2026

Abstract

Pretrained transformers such as BERT (Devlin et al., 2019) have been shown to be effective in many natural language tasks. However, they are under-explored for character-level sequence-to-sequence tasks. In this work, we investigate pretraining transformers for the character-level task of morphological inflection in several languages. We compare various training setups and secondary tasks where unsupervised data taken directly from the target task is used. We show that training on secondary unsupervised tasks increases inflection performance even without any external data, suggesting that models learn from additional unsupervised tasks themselves—not just from additional data. We also find that this does not hold true for specific combinations of secondary task and training setup, which has interesting implications for unsupervised training and denoising objectives in character-level tasks.

1 Introduction

Transformers have been shown to be an effective architecture for various natural language processing tasks (Vaswani et al., 2017), facilitating the ubiquitous method of pretraining on some unsupervised task with an abundance of data and then finetuning to a specific supervised task. Transformers have also been shown to be an effective architecture for character-level tasks such as grapheme-to-phoneme conversion (G2P) and morphological inflection (Wu et al., 2021).

However, very little work has explored the application of pretrained models to character-level tasks, which likely require different inductive biases than the more semantically-oriented tasks where pretraining is typical. For instance, Xue et al. (2022, ByT5), a multilingual pretrained transformer using byte inputs, showed impressive performance on several semantically-oriented benchmarks, as

well as on some character-level tasks including morphological inflection. However, it still under-performs the best two shared task submissions for the inflection benchmark (Vylomova et al., 2020).

The computational morphology community is frequently interested in low-resource languages—languages that do not have sufficient data available to apply standard NLP techniques. This is harder for morphologically complex languages, where the large set of inflectional patterns lead to an explosion in possible words, which become difficult to model with a small dataset. For these reasons, there is interest in building tools to aid in expanding morphological resources for language education tools, research, and documentation. Using NLP methods to build systems for analyzing and applying morphology in generalizable way to unseen words is thus a useful goal. Several shared tasks have been held to this end (Cotterell et al., 2016, 2018; Vylomova et al., 2020; Pimentel et al., 2021; Kodner et al., 2022), where a machine learning model that performs well can be seen as competently representing the underlying system of morphology for a given language.

In this work, we explore utilizing secondary unsupervised tasks—tasks similar to language modeling which can serve as auxiliary tasks in a multi-tasking setup or pretraining tasks in a pretraining setup—when training encoder-decoder transformers for the task of morphological inflection. We investigate the benefits of pretraining (PT) beyond expanding the vocabulary distribution during training and also compare it to multi-task learning (MTL). Following Kann and Schütze (2017), we use autoencoding (AE) as an unsupervised secondary task and additionally compare it to the denoising task of character-level masked language modeling (CMLM) (Wiemerslage et al., 2023; Devlin et al., 2019). We explore these methods in data-scarce settings to investigate their potential impact in the low-resource setting. Our data samples and code[1] are available publicly.

We specifically investigate the following research questions:

- **RQ1**: Is training on secondary unsupervised tasks an effective method for low-resource inflection, even without introducing any new words to the dataset? This allows us to measure the impact that unsupervised tasks have on a model outside of the obvious benefit of increasing data diversity.
- **RQ2**: Are denoising tasks a better alternative to autoencoding for morphological inflection?
- **RQ3**: When training a model for the given target task, does multi-task learning outperform pretraining?

Our results show that both unsupervised PT and MTL are effective for morphological inflection, even with samples prepared exclusively from the supervised data itself. We find that simply autoencoding the training words is more effective than CMLM in these data-scarce settings. Though the best method on average seems to be MTL with AE in our experiments, this is not consistent

[1] https://github.com/Abhishek-p/inflection-unsupervised-tasks

across every language. We also find that, in the MTL setup, CMLM actually performs worse than the baseline—though this is quickly reversed if we use out-of-distribution data for the secondary task.

2 Background Work

2.1 Character-level Sequence-to-sequence Tasks

Character-level sequence-to-sequence tasks, sometimes referred to as character transduction tasks, are a special case of neural sequence-to-sequence learning problems that deal with approximately word-sized sequences. They are characterized by small vocabularies $|\Sigma|$ and short source and target strings. Given source strings $S \in \Sigma^*$, target strings $Y \in \Sigma^*$, and optionally some features r to condition on, the goal of this task is to learn a mapping

$$f(S, r) \rightarrow Y \tag{1}$$

where $f(\cdot)$ is typically parameterized by a neural network. In this work, we focus on morphological inflection: a character-level task where a particular $s \in S$ is typically a lemma, $r \in R$ is a bundle of tags specifying inflectional features, and $y \in Y$ is a surface word of the lemma that expresses the specified morphological features, e.g.:

$$f(\text{cry}, \text{PST}) \rightarrow \text{cried} \tag{2}$$

Morphological inflection is an active area of research in NLP. Many shared tasks in the computational morphology community (Cotterell et al., 2017; Goldman et al., 2023) have spurred progress on this task, which can be considered a good proxy for measuring the extent to which machine learning models can acquire the system of morphology in a language. Wu et al. (2021) trained a transformer (Vaswani et al., 2017) for several character-level transduction tasks resulting in state-of-the-art results. We follow their training methodology for inflection models as our baseline in this work.

2.2 Transfer Learning

Additional data for tasks different from the target task can be used to learn representations that benefit some target task via transfer learning. This often entails training on an unsupervised secondary task like language modeling, due to the large availability of unannotated text and the high cost of attaining annotations for specific target tasks. There has also been a great deal of research in transfer learning with supervised tasks (Bingel and Søgaard, 2017; Phang et al., 2018; Pruksachatkun et al., 2020).

We explore two different setups for this, both of which are unsupervised. **Multi-task learning** (Caruana, 1997, MTL) refers to training some task(s) together with the target task by including samples from both in a single training run and combining the loss from each (Luong et al., 2016). Intuitively, a well-chosen secondary task will benefit the target task by encouraging a model to

learn a representation that minimizes the loss for both tasks simultaneously (Fifty et al., 2021). **Pretraining** (PT) refers to an alternative training setup in which models are first trained solely on secondary task(s) to encourage learning representations independent of the target task and then finetuned to some target task (Peters et al., 2018). Though both setups are similar, MTL relies on the joint optimization of multiple objectives, requiring a model to resolve all tasks at the same time. On the other hand, PT attempts to learn a representation that can be finetuned to a task later, by way of leveraging general encodings, or drawing upon an inductive bias learned in the pretraining phase.

2.3 Transfer Learning for Character-level Tasks

Kann and Schütze (2017) investigated the effectiveness of AE in an MTL setup by autoencoding with additional out-of-distribution words along with the target inflection task. Recently, Wiemerslage et al. (2023) pretrained various neural models on a character-level masked language modeling (CMLM) task, which follows the objective from Liu et al. (2019, RoBERTa), finding it can increase robustness to noise in the training data without the addition of new words. We follow them and use CMLM as the denoising task in our experiments. Similarly, Dong et al. (2022) pretrained a transformer encoder with a grapheme-based masking objective before finetuning to a downstream grapheme-to-phoneme (G2P) task and showed improvements for some datasets (Ashby et al., 2021).

2.4 Data Diversity and Multi-task Learning

Denoising methods involve adding some noise to an input and then decoding the original form as it was before the noise step (Vincent et al., 2010), e.g.:

$$\text{tr@e@} \rightarrow \text{tried}, \tag{3}$$

where @ is a noise token that is applied in a data preprocessing step, and which the model must learn to replace with the original token. Many denoising strategies have been proposed for pretraining language models (Devlin et al., 2019; Raffel et al., 2019; Lewis et al., 2020), which may have advantages for particular downstream tasks.

Martínez Alonso and Plank (2017) showed that token distribution for data in an auxiliary task has been shown to have a strong impact on performance. In an exploration of supervised secondary tasks, Bingel and Søgaard (2017) found that, when training with MTL for many NLP tasks, the out-of-vocabulary rate in the auxiliary task is positively associated with performance. This can also translate to unsupervised training for character-level tasks, where external data can positively impact model training regardless of the task for training on that data. Bjerva et al. (2019) perform MTL on many supervised tasks annotated for the same input examples. They train on the predictions for auxiliary tasks on the test set in a transductive learning setup, which increases performance. Krishna et al. (2023) found reusing downstream task data for unsupervised

pretraining—which they refer to as self pretraining—to be an effective alternative to pretraining on external data. In experiments, they show that this often outperformed finetuning an off-the-shelf model that was pretrained on external data.

Similarly, in this work, we explore how data diversity impacts performance. That is, we compare secondary task words drawn from the target task to external data. This isolates secondary task impact from the effect of increased data diversity.

3 Architecture and Training

In this section we discuss our training methodology including architecture, training setups, and tasks.

3.1 Architecture

All of our experiments utilize the character encoder-decoder transformer from Wu et al. (2021). We use 4 encoder and 4 decoder layers, 4 attention heads, embedding size 256, and a feed-forward layer with hidden size 1024. We also follow their methodology for selection of the best checkpoint, where the highest accuracy on a validation set is selected out of 50 checkpoints. For all hyperparameters, refer to Wu et al. (2021).

3.2 Training Tasks

Morphological Inflection In this work, morphological inflection is the only supervised task considered, and it is the target task for all experiments. We formulate the inflection task identically to prior work (Kann and Schütze, 2016; Wu et al., 2021).

CMLM We follow Wiemerslage et al. (2023) in implementing CMLM for the denoising secondary task, where masking hyperparameters follow RoBERTa, though we increase the mask sampling rate. Specifically, we sample $m = 20\%$ of all input characters for masking. Then, for each character, with probability $p_m = 0.8$ we replace it with a special mask token, with probability $p_r = 0.1$ we replace it with another character randomly sampled from the vocabulary, and with probability $p_k = 0.1$ we leave the character unchanged.

AE We additionally compare to autoencoding as a secondary task, in which we do no denoising at all: the source and target word are identical.

3.3 Training Setups

We compare three different training setups: supervised-only, pretrain-finetune (PT) and multi-task learning (MTL).

Pretrain-Finetune (PT) We first pretrain an encoder-decoder model on an unsupervised secondary task and then train it on supervised data in a finetuning stage. We train the encoder-decoder fully in both the pretraining and

finetuning stages. The finetuning stage is nearly identical to the supervised training setup, except we train from a pretrained checkpoint instead of training from scratch. We train both stages for 800 epochs. Since this is a two-stage setup, we apply model selection criteria twice. In the pretraining stage, the best checkpoint is chosen by minimizing evaluation loss on the secondary unsupervised task. This means that in the pretraining stage the model is motivated to learn representations over the character sequences from the vocabulary. The finetuning stage model selection remains identical to the supervised setup.

Multi-task Learning (MTL) Similar to the setup in Kann and Schütze (2017), models are trained simultaneously for the target task and an unsupervised secondary task. We assign a fixed task weight factor α for the unsupervised secondary task and β for the target inflection task. For all experiments, we set $\alpha = 1$ and $\beta = 1$, and compute loss as the weighted sum of the two:

$$\mathcal{L}(\theta) = \alpha \cdot \ell_1(z_i, g(z_i; \theta)) + \beta \cdot \ell_2(x_i, f(x_i, t_i; \theta), y_i) \quad (4)$$

where f is the inflection task as in Section 2.1, $g(\cdot)$ is the unsupervised secondary task function, $z \in Z$ and $o \in O$ are the unsupervised source and target, and ℓ_1 and ℓ_2 are loss functions for the two tasks, respectively. In initial experiments, we tried varying the task weights and found little impact on performance.

Although the training objective is to minimize $\mathcal{L}(\theta)$, the best model is selected as in the previous setups with the best evaluation accuracy on the target task after training for 800 epochs. We added specific task identifiers (i.e., [TASK1], [TASK2]) to the input during training and inference. These identifiers are part of the input, however separated from the source (and features) with a start token. This way the model can identify the relevant task for the sample.

Supervised-only This is identical to the training setup from (Wu et al., 2021), where a model is trained only for the morphological inflection task. We follow them in training the model on the target-task data for 800 epochs and the best of 50 checkpoints by validation accuracy is chosen.

4 Data

4.1 Target-task Data

Morphological inflection training data is sampled from the 2023 shared task on morphological inflection (Goldman et al., 2023). This supervised dataset consists of triples comprising (lemma, feature set, inflected form).

It consists of 10k train samples and 1k each of development and test samples for 26 languages and an additional unvocalized variant (heb_unvoc) of Hebrew (heb). We differentiate Hebrew variants in our experiments and results, although we refer to it collectively as a language. In order to simulate a data-scarce setting, we randomly subsample the train split to 1k samples, as in the medium setting of the SIGMORPHON 2017 shared task (Cotterell et al., 2017). We also flatten the hierarchical features following most submissions to the 2023 shared task. This is performed by parsing the features during pre-processing

and combining the multi-level features with special characters to make combined features. Consequently, our task data consists of the development and test splits and a subsampled 1k train split, all with flattened features.

We inherit the fact that the shared task partitions lemmas between the 3 splits, which means all experiments require generalizing to unseen lemmas.

4.2 Extracted Data

We experiment with secondary-task data taken exclusively from the training data. That is, given a labeled triple from the supervised morphological inflection dataset like (debut, V;PRS;NOM(3,SG), debuts), we make two unsupervised training samples: debut → debut and debuts → debuts.

4.3 External Data

We perform an additional analysis with data sampled from a source external to the supervised data, which we refer to as external data. Here, we sample words from the universal dependencies (UD) treebanks (Zeman et al., 2023). Since the availability of languages in UD does not directly correspond to the 2023 shared task data, we select 19 languages for which treebanks are available. The specific treebank used for dataset creation for each language is mentioned in Table **??**. From each language's treebank, we sample 2k words to use for secondary tasks. For details on how words are sampled, see appendix (Section A.3).

5 Experiments

5.1 Experimental Setup

We compare five model variants: **baseline** refers to the supervised model following Wu et al. (2021). We refer to **PT-CMLM** for models pretrained on the extracted data with the CMLM objective and then finetuned to the supervised data, whereas **MTL-CMLM** models train both tasks in MTL setup. **PT-AE** and **MTL-AE** reflect the same respective training setups, but use autoencoding as the secondary task. With these variants, we can compare all models to the baseline to answer RQ1, and we can compare across training setups and secondary tasks to answer RQ2 and RQ3, respectively.

5.2 Results

In Table **??** we present the main results: the accuracy of all five model variants averaged over all 27 languages on each of the development and test set. For a per-language results breakdown, see Table **??**. For all comparisons, we focus on average accuracy on the test set.

The baseline is outperformed by almost all model variants that have been trained on secondary tasks. This means that secondary unsupervised tasks are beneficial even when no new data is introduced (RQ1). PT-CMLM outperforms

the baseline by 1.84 absolute accuracy, only performing worse than the baseline on 6 languages: deu, ita, jpn, rus, sme, sqi. PT-AE performs even better, outperforming the baseline by 3.16 absolute accuracy, but performs worse than the baseline in 5 languages: bel, dan, jpn, mkd, rus. We perform a paired permutation test and find all comparisons to the baseline to be statistically significant ($p < 0.03$).

A comparison across unsupervised objectives shows that AE outperforms CMLM (RQ2). Although on average the difference is small (1.32) in the PT setup, AE outperforms CMLM substantially by 10.9 absolute accuracy in the MTL setup on the test set. Overall, MTL-AE is the best performing model, which indicates that MTL is a better setup for this task than PT (RQ3). However, this is not true when using the denoising objective. Only on 6 languages (dan, fra, heb, heb_unvoc, klr, san) does MTL-CMLM outperform the baseline, and on average it performs worse than the baseline.

Unsupervised Training on the Target-task Data: Most of the models outperform the baseline using strictly extracted finetuning data for unsupervised training with no additional words. This indicates that unsupervised tasks are effective for transfer learning in low-resource scenarios separately from the effect of exposing the model to new data. For PT, we hypothesize that the unsupervised pretraining task imparts some inductive bias to the model related to capabilities that are crucial to the downstream task. For example, learning a strong bias towards copying characters, which is a common operation in morphological inflection, or learning a strong language model over the character sequences in the training data, before learning to condition on features.

Although MTL-AE consistently performs best, the MTL setup performs very poorly with CMLM unlike in the PT setup. This indicates that learning from secondary tasks functions drastically different between PT and MTL, where MTL is perhaps more sensitive to the choice of secondary task. We explore this in more depth in Section 6.

AE Is Unreasonably Effective: Given the simplicity of the autoencoding task and the fact that we do not introduce any new data beyond the finetuning dataset, this large increase in accuracy implies a surprising capacity for learning that has not been previously explored.

6 When Does Denoising Hurt MTL?

There is a remarkable gap in performance between MTL-CMLM and PT-CMLM (6.29 absolute accuracy) as well as MTL-AE (10.9 absolute accuracy). While denoising is a useful objective to pretrain on, it actually hurts performance in an MTL setup in our experiments. This also begs the question: why is AE a valid secondary task when multitasking (our best overall setup), but not denoising? We hypothesize that denoising negatively impacts model learning because it is a sufficiently different task optimized on the same words as inflection. In the PT setup, if denoising learns a representation that conflicts with the finetuning task, this can be resolved by optimizing strictly on the finetuning task in a second

phase. However, perhaps when optimizing jointly, the denoising objective skews the model distribution for the training words. This would imply that if denoising is done on external data, it should not have such a negative impact.

Based on these initial highly negative results for MTL-CMLM, we perform an additional analysis to investigate the impact of data diversity on both secondary tasks in an MTL setup.

Here all data for unsupervised learning is sampled from a source external to the finetuning data. We use Universal dependencies (Zeman et al., 2023, UD) as the source of external data, which we discuss in more detail in Subsection A.3.

Universal Dependencies Data: All inflection task data (Subsection 4.1) is derived from the SIGMORPHON 2023 shared task, which samples its splits from UniMorph (Batsuren et al., 2022)—a type-level multilingual morphological resource for NLP, with labeled morphological paradigms comprising 182 languages, 122M inflections, and 769K derivations extracted semi-automatically. Universal Dependencies is another multilingual NLP resource consisting of treebanks in 148 languages (as of the 2.13 release), though annotated data comprises token-level corpora. We choose UD as the source of external data in order to simulate a more naturally occurring type distribution than UniMorph. Whereas UniMorph types are likely to (i) be of the same part of speech as the test set, and (ii) represent interesting inflections that may be rare in a realistic low-resource scenario, UD contains types more representative of any arbitrary text. At the same time, unlike raw text scraped from the internet, UD data is relatively clean and has been vetted by experts, which ensures we do not experiment with e.g., data that has been misidentified as the target language or is otherwise contaminated.

Since not all 27 languages have treebanks in UD, we manually select a single treebank in only 19 of the 27 languages for these experiments. All models that use external data for secondary tasks are referred to with the suffix "-UD".

6.1 Results

In Table **??**, we present results for all 19 languages where MTL-CMLM-UD and MTL-AE-UD use external data sampled from UD for the respective secondary task. Using external data results in a 13.24 increase in absolute accuracy over MTL-CMLM, and outperforms the baseline substantially. On the other hand, the external data also leads to improved performance for MTL-AE-UD, but at a much smaller scale of 3.38 absolute accuracy over MTL-AE. On average, MTL-AE and MTL-CMLM-UD perform similarly. In a paired permutation test, all results have a statistically significant increase in performance over the baseline, except for MTL-CMLM which underperforms the baseline ($p < 0.006$). We now focus on the substantial increase for MTL-CMLM-UD. This result supports the hypothesis that jointly optimizing a sufficiently different task from the target task, but on the same data causes issues. Consider the MTL-CMLM-UD model. The denoising task is learning representations over character sequences that are different from those in the target task, allowing the two tasks to up-

Figure 1: The distribution of secondary task gradients between 20% and 30% training as in Bingel and Søgaard (2017) for cases in which the target task gradients are > 0. A negative number indicates the model is still improving upon the secondary task.

date model parameters for separate distributions, and reducing conflicts in the joint-optimization. Indeed substituting the extracted data with external data when using the same denoising task leads to a remarkable improvement in performance.

6.2 Training Dynamics in MTL

We analyze the training dynamics between both the target and secondary task to further explain the MTL behavior. Bingel and Søgaard (2017) find that features of the learning curves are strong predictors of which secondary tasks lead to the best performance in an MTL setup. They hypothesize that MTL helps most in cases where a target task converges quickly, while the secondary task is still learning, which may help target tasks avoid getting stuck in local minima. We explore this hypothesis by, like them, looking at the gradients of each task's training loss with respect to epochs, where the losses are recorded at the end of each epoch.

We then check the target task gradients that are > 0 within the first 20%–30% of training epochs, which we can consider to indicate that the task is plateauing early in training. In Figure **??** we provide violin plots of the secondary task gradients for those early target task plateaus in Sami—the language with the highest MTL improvement when UD data is added, and Danish—the language with the lowest improvement. For both languages, AE distributions have small variance around 0, whereas the CMLM plots show wider distributions. This reflects the fact that the CMLM loss is less stable, oscillating much more than the AE loss. More directly addressing the hypothesis about helping the target task recover from local minima, we see distributions that are either top-heavy, or normal for Danish, where no secondary task leads to a very large increase in performance over the baseline. On the other hand, the CMLM-UD distribution is more bottom-heavy for Sami, indicating that there are more negative gradients, and thus more epochs where the model is still learning this task when the target task seems to plateau. The AE distribution, while still low variance around 0, also have lower negative gradients compared to Danish.

This small analysis suggests two things. First, we have weak support for the hypothesis that MTL helps when the secondary task continues to converge when the target task plateaus early. We see more negative values in the Sami distribution where MTL is more helpful, especially in the CMLM-UD secondary task when compared to the CMLM without UD data. Second, AE, typically the best secondary task in our experiments, appears to have a lower variance in gradients, indicating that the training loss is more stable. Indeed, the variance for CMLM gradients is larger in Sami, where CMLM hurts performance, and the variance is smaller in Sami when we add the UD data, which has a large positive impact.

7 Conclusion

In this work, we explored multiple methods for transfer learning for morphological inflection, many of which showed remarkable performance for a large set of languages. We investigated two different training methods: pretraining-finetuning and multi-task learning, and two different secondary tasks: denoising and autoencoding. In a low-resource setting, we found that secondary unsupervised tasks are effective even without the addition of any new vocabulary items beyond the finetuning dataset. While pretraining is an effective setup for improving morphological inflection without any external data, multi-task learning with an autoencoding objective is the best setup in all experiments. On the other hand, multi-task learning with the CMLM denoising objective is the worst performing setup, performing below the baseline on average. In further analysis, we found that performing CMLM on external data that is separate from the finetuning data solves this issue, resulting in significantly better performance.

The success of denoising objectives such as MLM cannot be denied for large-scale training and semantically oriented tasks. Our experiments and results show that similar tasks are effective in data-scarce settings for character-

level tasks like morphological inflection. In practice, it seems that low-resource character-level tasks should always consider training in a multi-task setup with an autoencoding secondary task even if the supervised training data is the only available data—and exploring denoising objectives if unsupervised data from an external source is available.

8 Future Work

The denoising tasks requires hyperparameters for the instrumentation of the noise. Due to this, further work is required in exploring these tasks under different hyperparameter settings with multiple methods to shed light on their sensitivity and ability to improve models for character-level tasks such as morphological inflection and G2P. Future work should also consider exploring more secondary tasks, especially based on particular morphological phenomenon in diverse languages.

Limitations

- Our work is limited to the character-level task of morphological inflection. Thus, findings may not hold for other similar tasks such as G2P and interlinear glossing.
- Considering the sensitivity of training methods to vocabulary and data sizes, it is unclear whether these results can be extrapolated to different scenarios.
- Our work does not explore the disparity of performance of the methods across languages and requires expert analysis over various linguistic features.

Acknowledgments

We thank the anonymous reviewers for their useful suggestions and feedback and the NALA Lab at the University of Colorado Boulder. This work utilized the Blanca condo computing resource at the University of Colorado Boulder. Blanca is jointly funded by computing users and the University of Colorado Boulder.

A Data details

A.1 Limitations of UniMorph and SIGMORPHON

The UniMorph project is the primary source for the dataset. It draws heavily from Wiktionary[2] in a semi-automated way based on Kirov et al. (2016). Wiktionary is a collaboratively built resource which, despite processes to promote accuracy, is not a linguistic resource that is considered as gold-standard data. The semi-automated methodology, sources, and broad mandate limits the utility and effectiveness of the dataset. A notable example is Ahmadi and Mahmudi (2023), which discusses this in the context of Sorani (ckb) also known as Central Kurdish (not one of the 27 languages in this work). The limitations of the dataset used in this work, being only very recently released, are not well-studied, and consequently also apply to our work.

A.2 Selection and Sampling

Many features of morphological inflection data, such as overlap and frequency, have been shown to be important factors for model performance (Kodner et al., 2023). (Muradoglu and Hulden, 2022) demonstrated how data could be sampled using active learning methods to improve model performance. Since we investigate training methods rather than data methods, we perform analysis on data which has been selected specifically for benchmarking purposes. We recommend the readers check Section 4 "Data preparation" of the shared task paper Goldman et al. (2023) for more information on the data methods used for target-task data selection and splits. We discuss details relevant to our selection and sampling below.

Lemma Overlap: The 2023 shared task dataset was specifically designed to prevent lemma overlap between any of dev, train, and test. Since we only subsample from train, the lack of lemma overlap is maintained in our datasets, and is thus not a relevant point of analysis as in other work (e.g. Kodner et al., 2023).

A.3 Preparing Additional Data from UD Treebanks

With a fixed seed, we randomly sample words from the selected UD Treebank to prepare an unlabeled training set of size 2k for each language. We perform sampling only after filtering out NUM and PUNCT tagged and tokenized words (Nivre et al., 2020). We do not otherwise use the token-level annotations from UD, simulating a more realistic data setting than the one UniMorph words represent.

Table **??** shows the 19 languages from the shared task for which UD was used for additional training data in our investigation of the denoising task in the MTL setup. We list the specific treebanks used in order to encourage reproducibility. We preserve both the data and corpus information for the selected

[2] https://www.wiktionary.org/

words. Specifically, we have also collected the token frequency, UPOS frequency, and character frequency for each of the additional data sampled, to be made available with the code for future analysis.

B Models and Experimental Details

B.1 Implementation

All models are implemented with a fork of yoyodyne[3], which is built over pytorch-lightning (Falcon and The PyTorch Lightning team, 2019). We utilize yoyodyne's existing implementation of the Wu et al. (2021) models. We additionally implemented the CMLM objective, two stage training for PT setup, and the MTL setup including data and loss combination using the framework.

B.2 Compute and Infrastructure

For reproducibility, we utilize only Nvidia V100 GPUs for our experiments. The reported models together required ∼180 hours of GPU time.

B.3 Reproducibility

In addition to using a consistent GPU architecture, we use a fixed random seed of 1 for all our model experiments. We also maintain copies of the specific data.

B.4 Morphological Inflection in Japanese

Organizers of the 2023 shared task note the challenges that Japanese presents in morphological inflection, namely due to its extremely large vocabulary size. In our work this persists as most models perform poorly on Japanese and do not meaningfully improve upon the baseline.

C Significance Testing

In order to analyze the significance of our results, we perform a paired permutation test between test accuracies of all the models compared to the baseline. For all these tests, we use the null-hypothesis that the mean difference between the test accuracies for these pairs is 0 and run the tests with 100k sampled permutations of the differences using SciPy (Virtanen et al., 2020).

[3] https://github.com/CUNY-CL/yoyodyne

Table 1: The 27 typologically diverse languages (Subsection 4.1) from the 2023 shared task, all of which are investigated in this work. We use some UD Treebanks for our analytical experiments in Subsection 6, the specific treebanks are listed in the final column.

ISO 639-2	Language	UD Treebank used
afb	Arabic, Gulf	Arabic-PADT
amh	Amharic	Amharic-ATT
arz	Arabic, Egyptian	
bel	Belarusian	Belarusian-HSE
dan	Danish	Danish-DDT
deu	German	German-GSD
eng	English	English-EWT
fin	Finnish	Finnish-FTB
fra	French	French-GSD
grc	Ancient Greek	Ancient_Greek-Perseus
heb	Hebrew	Hebrew-HTB
heb_unvoc	Hebrew, Unvocalized	
hun	Hungarian	Hungarian-Szeged
hye	Eastern Armenian	Armenian-ArmTDP
ita	Italian	Italian-ISDT
jpn	Japanese	Japanese-GSD
kat	Georgian	Georgian-GSD
klr	Khaling	
mkd	Macedonian	Macedonian-MTB
nav	Navajo	
rus	Russian	Russian-GSD
san	Sanskrit	Sanskrit-UFAL
sme	North Sami	North_Sami-Giella
spa	Spanish	Spanish-AnCora
sqi	Albanian	
swa	Swahili	
tur	Turkish	Turkish-IMST

Table 2: The development and test accuracies of the 5 model variants, for all the 27 languages. For each language, the highest development accuracy is underlined and highest test accuracy is bolded.

Language	ISO 639-2	Baseline		PT-CMLM		PT-AE		MTL-CMLM		MTL-AE	
		Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Arabic, Gulf	afb	68.8	65.7	72.2	71.9	73.0	68.8	65.7	61.3	76.55	73.66
Amharic	amh	44.6	42.9	42.5	39.7	43.0	41.9	36.7	28.6	44.0	40.7
Arabic, Egyptian	arz	82.8	82.5	79.9	78.1	80.0	78.9	43.3	33.8	80.9	81.8
Belarusian	bel	61.2	59.0	55.8	54.5	62.1	61.0	65.4	61.4	63.4	56.4
Danish	dan	81.7	80.1	62.3	61.5	68.0	65.7	72.2	61.7	70.0	60.4
German	deu	68.2	68.7	89.4	88.6	75.4	80.4	83.2	83.6	91.6	90.9
English	eng	91.6	88.2	92.0	91.8	94.3	90.3	74.3	73.2	89.5	89.5
Finnish	fin	74.6	71.2	88.1	87.1	87.8	86.6	92.3	90.9	81.4	78.5
French	fra	15.2	15.8	38.5	37.2	52.8	46.5	81.4	76.3	84.2	82.3
Ancient Greek	grc	54.1	56.7	72.7	71.7	77.6	75.2	91.9	88.9	89.7	85.9
Hebrew	heb	74.2	78.7	66.9	65.5	82.2	80.1	82.7	73.6	94.3	93.3
Hebrew, Unvocalized	heb_unvoc	81.5	85.1	84.6	86.7	79.3	72.2	85.8	74.1	44.1	42.8
Hungarian	hun	75.7	79.4	19.5	14.8	64.4	61.0	63.5	47.1	81.8	82.9
Eastern Armenian	hye	79.2	85.1	86.4	87.0	66.7	64.2	80.1	77.95	65.7	58.3
Italian	ita	90.5	88.2	80.3	79.2	88.1	89.0	74.8	64.2	74.8	66.3
Japanese	jpn	15.8	16.6	13.1	13.7	14.0	19.3	5.6	5.6	91.8	91.8
Georgian	kat	70.2	68.8	76.0	71.7	76.4	78.55	89.0	82.8	89.6	89.9
Khaling	klr	91.6	91.2	37.8	42.4	42.5	43.3	88.7	81.1	44.4	44.4
Macedonian	mkd	83.1	85.6	76.6	74.8	49.0	52.6	61.5	57.7	80.8	78.1
Navajo	nav	36.1	33.1	49.0	50.5	43.9	33.8	80.4	75.73	79.3	75.4
Russian	rus	78.7	80.7	55.0	56.5	55.8	56.5	83.7	82.5	91.4	93.0
Sanskrit	san	55.0	48.0	62.1	62.2	89.4	83.8	75.4	74.3	94.3	93.3
Sami	sme	57.3	62.1	88.2	86.2	75.4	73.2	76.3	72.7	44.1	42.8
Spanish	spa	88.2	85.0	93.5	94.5	92.0	90.9	90.4	90.4	80.1	82.9
Albanian	sqi	78.6	81.6	85.3	87.2	88.1	87.2	21.9	21.9	57.9	58.3
Swahili	swa	93.5	91.8	85.3	87.8	86.0	87.0	72.2	72.7	66.7	66.3
Turkish	tur	85.3	86.4	88.1	90.3	87.8	88.0	75.2	74.9	90.3	91.8
Average		71.75	67.21	73.84	69.05	74.14	70.37	65.75	62.76	76.55	73.66

Table 3: Results for our models by language from the experiments with external data, reporting development and test accuracy. For each language, the highest development accuracy is underlined and highest test accuracy is bolded. Note: results for non-UD models are identical to Table **??**.

Language	ISO	Baseline		MTL-CMLM		MTL-AE		MTL-CMLM-UD		MTL-AE-UD
	639-2	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev
Test										
Arabic, Gulf	afb	68.8	65.7	65.7	61.3	76.55	73.66	74.58	71.28	76.31
72.22										
Amharic	amh	44.6	42.9	36.7	28.6	44.0	40.7	78.09	74.66	78.09
74.66										
Belarusian	bel	61.2	59.0	65.4	61.4	63.4	56.4	72.8	72.7	72.8
72.7										
Danish	dan	81.7	80.1	62.3	61.5	68.0	65.7	74.9	61.0	74.9
61.0										
German	deu	68.2	68.7	89.4	88.6	75.4	80.4	66.6	65.3	66.6
65.3										
English	eng	91.6	88.2	92.0	91.8	94.3	90.3	62.2	62.2	62.2
62.2										
Finnish	fin	74.6	71.2	88.1	87.1	87.8	86.6	83.7	82.9	83.7
82.9										
French	fra	15.2	15.8	38.5	37.2	52.8	46.5	75.4	76.3	75.4
76.3										
Ancient Greek	grc	54.1	56.7	72.7	71.7	77.6	75.2	91.9	88.9	91.9
88.9										
Hungarian	hun	74.2	78.7	66.9	65.5	82.2	80.1	82.7	73.6	82.7
73.6										
Hebrew	heb	81.5	85.1	84.6	86.7	79.3	72.2	85.8	74.1	85.8
74.1										
Eastern Armenian	hye	79.2	85.1	86.4	87.0	66.7	64.2	80.1	77.95	80.1
77.95										
Italian	ita	90.5	88.2	80.3	79.2	88.1	89.0	74.8	64.2	74.8
64.2										
Japanese	jpn	15.8	16.6	13.1	13.7	14.0	19.3	5.6	5.6	5.6
5.6										
Russian	rus	70.2	72.1	76.0	71.7	76.4	78.55	89.0	88.7	89.0
88.7										
Sanskrit	san	91.6	91.2	37.8	42.4	42.5	43.3	32.2	32.2	32.2
32.2										
Sami	sme	83.1	85.6	76.6	74.8	49.0	52.6	80.1	81.8	80.1
81.8										
Spanish	spa	55.0	48.0	62.1	62.2	89.4	83.8	57.9	58.3	57.9
58.3										
Turkish	tur	78.6	81.6	85.3	87.2	88.1	87.2	74.8	66.3	74.8
66.3										
Average		69.51	64.39	62.45	58.98	74.58	71.28	76.31	72.22	78.09
74.66										