SigMoreFun Submission to the SIGMORPHON Shared Task on Interlinear Glossing

Taiqi He [*], **Lindia Tjuatja** [*], **Nate Robinson**,
Shinji Watanabe, **David R. Mortensen**, **Graham Neubig**, **Lori Levin**
Language Technologies Institute
Carnegie Mellon University
{taiqih,ltjuatja,nrrobins,swatanab,dmortens,gneubig,lsl}@cs.cmu.edu

Abstract

In our submission to the SIGMORPHON 2023 Shared Task on interlinear glossing (IGT), we explore approaches to data augmentation and modeling across seven low-resource languages. For data augmentation, we explore two approaches: creating artificial data from the provided training data and utilizing existing IGT resources in other languages. On the modeling side, we test an enhanced version of the provided token classification baseline as well as a pretrained multilingual seq2seq model. Additionally, we apply post-correction using a dictionary for Gitksan, the language with the smallest amount of data. We find that our token classification models are the best performing, with the highest word-level accuracy for Arapaho and highest morpheme-level accuracy for Gitksan out of all submissions. We also show that data augmentation is an effective strategy, though applying artificial data pretraining has very different effects across both models tested.

1 Introduction

This paper describes the SigMoreFun submission to the SIGMORPHON 2023 Shared Task on interlinear glossing (Ginn et al., 2023). Given input text in a target language, the task is to predict the corresponding interlinear gloss (using Leipzig glossing conventions). IGT is an important form of linguistic annotation for the morphological analysis of languages, and also serves as an extremely valuable resource for language documentation and education for speakers of low-resource languages.

There were two tracks for this shared task, Track 1 (closed) and Track 2 (open). For Track 1, systems could only be trained on input sentences and glosses; in Track 2, systems could make use of the morphological segmentation of the input as well as any (non-IGT) external resources. Since the Track 2 setting better matches the long-term research goals of our team, we only participate in this open track.

In our submission, we investigate two different approaches. First, we attempt data augmentation by either creating our own artificial gloss data by manipulating the existing training data, or by utilizing existing resources containing IGT in other languages (§2). Second, we explore two different models for gloss generation (§3). The first builds off the token classification baseline, while the second uses a pretrained multilingual seq2seq model.

Finally, we also attempt to post-correct model outputs with a dictionary. We apply this to Gitksan and find that this, combined with our other approaches, results in the highest morpheme-level accuracy for Gitksan in Track 2.

2 Data Augmentation

One major challenge for this shared task is the scale of data provided. All of the languages have less than 40k lines of training data, and all but Arapaho have less than 10k. The smallest dataset (Gitksan) has only 31 lines of data. Thus, one obvious method to try is data augmentation. More specifically, we try pretraining our models on different forms of augmented data before training them on the original target language data.

We explored two forms of data augmentation. First, we generated artificial gloss data in the target language by permuting morphemes in the existing training data. Second, we utilized data from the Online Database for Interlinear Text (ODIN; Lewis and Xia, 2010; Xia et al., 2014) to see if transfer learning from data in other languages can help improve performance.

2.1 Artificial Data

A challenge our team faced with respect to data augmentation is figuring out how to obtain additional data when we do not have much knowledge of the languages' grammatical systems, along with

[*] These authors contributed equally

the fact that these languages are generally from digitally under-resourced language families. Furthermore, we wanted our solution to be easily implemented and relatively language agnostic due to time constraints and practical usability for researchers working on a variety of languages.

Thus, one avenue of data augmentation we tried was to create artificial data from the provided training data. This requires no rule-writing or knowledge of the grammar of the language, and thus could be applied quickly and easily to all of the languages in the shared task.

We used a naive word-swapping method to randomly swap morphemes that occur in similar contexts to create new sentences. To do this, for each gloss line, we replace each word stem (that has a gloss label affix) with "STEM" to create a skeleton gloss. We naively determine if a label is a stem by checking if it is in lowercase. We do not do this to words that do not have affixes as (with the exception of Uspanteko) we do not have access to parts of speech, and do not want to swap words that would create an ungrammatical sequence.

We create a dictionary mapping each skeleton word gloss to possible actual glosses, and map each actual gloss to possible surface forms (we make no assumptions that these mappings are one-to-one). We then randomly sample k random skeleton glosses (in this case, we used k equal to roughly three times the amount of training data) and randomly fill in words that match the format of skeleton words present in the line.

(1) to (3) below illustrate an example in this process. We create a skeleton gloss (2) from the Gitksan sentence in (1) by replacing the all word stems that have an affix with "STEM" in both the segmentation and gloss tiers—in this case, only *'witxw-it* applies to this step. Then to create the artificial data in (3), we replace the skeleton word and corresponding gloss with another word from the training data that has the same skeleton form, in this case *hahla'lst-it*.

(1) ii nee-dii-t naa dim 'witxw-it20
CCNJ NEG-FOC-3.I who PROSP come-SX

(2) ii nee-dii-t naa dim STEM-it
CCNJ NEG-FOC-3.I who PROSP STEM-SX

(3) ii nee-dii-t naa dim hahla'lst-it
CCNJ NEG-FOC-3.I who PROSP work-SX

While this method may create a somewhat unnatural input surface sequence (as we are unable to capture phonological changes in the surface form and corresponding translations may be nonsensical), this method guarantees that the structure of the gloss is a naturally occurring sequence (as we only use gloss skeletons that are present in the input). However, a limitation of this method is that it does not extend to out-of-vocabulary tokens or unseen gloss structures. Furthermore, as we cannot generate a gold-standard translation for the artificial data, we do not make use of a translation in training.

2.2 ODIN

Another potential avenue for data augmentation is transfer learning from data in other languages, which has been shown to be an effective method to improve performance in low-resource settings (Ruder et al., 2019).

The available resource we utilize is ODIN, or the Online Database for Interlinear Text (Lewis and Xia, 2010; Xia et al., 2014). ODIN contains 158,007 lines of IGT, covering 1,496 languages.

We use the 2.1 version of ODIN data and convert the dataset to the shared task format, and filter out languages with fewer than five glossed sentences. However, there remains significant noise in the dataset that could cause significant alignment issues for the token classification models. Therefore we opt to only train the ByT5 models on ODIN, in the hope that this model is less sensitive to alignment errors. Indeed, we find that the ByT5 model finetuned first on ODIN receives a performance boost when finetuned again on the shared task data.

3 Models

We explore two models for gloss generation. The first one is built upon the token classification baseline with some improvements, and we treat this model as our internal baseline. The second model we deploy tests whether we can achieve competitive performance by finetuning a pretrained character based multilingual and multitask model, ByT5. For this model, we perform minimal preprocessing and use raw segmented morphemes and free translations if available.

3.1 Token Classification Transformer

We use the baseline Track 2 model provided by the organizers as a starting point. The original implementation randomly initializes a transformer model from the default Huggingface RoBERTa base con-

figuration, and uses a token classification objective with cross-entropy loss, where each gloss is treated as a distinct token. The morphemes and free translations are tokenized by space and dashes, with punctuations pre-separated. They are concatenated and separated by the SEP token and are used as the inputs to the model. We modify the original Track 2 baseline model to obtain a better baseline. We use pretrained weights from XLM-RoBERTa (XLMR) base (Conneau et al., 2020), instead of randomly initializing the weights. We also slightly modify the morpheme tokenizer to enforce that the number of morpheme tokens matches the number of output gloss tokens exactly.

Additionally, we introduce the COPY token to replace the gloss if it matches the corresponding morpheme exactly. An example from Natugu is shown in gloss (4):

(4) 67 . mnc-x Mzlo Skul
COPY COPY be-1MINI COPY COPY

We believe this would improve performance by removing the need to memorize glossed code-switching and proper nouns, though it is only effective if the code-switched language is the same as the meta language (e.g. English code-switching in the Arapaho data), and would have no effect if the source language uses a different orthography or is code-switched to another language, where the gloss would not matched the morpheme form exactly. This method also compresses all punctuation markers into one token, but the usefulness of this side effect is less clear.

Since we are using pretrained weights, it is then natural to explore integrating the pretrained tokenizer. Since XLMR was not trained on any of the source languages, it makes the most sense to only use the pretrained tokenizer to tokenize free translations, if they are available, and extend the vocabulary to include morphemes.

3.2 Finetuned ByT5

Multi-task and multi-lingual pretrained large language models have been shown to be effective for many tasks. We explore whether such models can be used effectively for glossing. We conduct experiments with both mT5 (Xue et al., 2021) and ByT5 (Xue et al., 2022), but ByT5 is preferred because it takes raw texts (bytes or characters) as inputs and in theory should be more effective for unseen languages. Following the text-to-text format inherited from T5 (Raffel et al., 2020), we use a prompt based multilingual sequence to sequence objective for both models. The prompt template is: "Generate interlinear gloss from `[source language]`: `[segmented morphemes]` with its `[meta language]` translation: `[free translation]` Answer: ". Data from all languages are mixed together and shuffled, with no up or down sampling. After initial experiments, we find ByT5 outperforms mT5 across all languages, and therefore we only conduct subsequent experiments on ByT5 and report those results.

Upon initial experiments, we also find the results for Lezgi to be lower than expected. We hypothesize that the fact that the data are in Cyrillic script causes this deficiency, since ByT5 was trained on far less Cyrillic data than data in the Latin script. Therefore we create an automatic romanization tool, sourced from Wikipedia[1] and integrated in the Epitran package (Mortensen et al., 2018), and convert all Lezgi data to Latin script for ByT5 finetuning.

After inspecting the outputs of the ByT5 models, we find cases where punctuations are attached to the previous glosses, instead of being separated by a space as is standard in the training sets. This is probably due to the fact that the model was pretrained on untokenized data and this behavior is preserved despite finetuning on tokenized data. We therefore use a simple regular expression based tokenizer to fix the inconsistencies. We notice that the procedure only gives performance boost on Gitksan, Lezgi, Uspanteko, and Natugu, and so we only apply the procedure to those languages, leaving the rest of the outputs unchanged.

4 Dictionary Post-correction: Gitksan

One of the key challenges for extremely low resource languages is the integration of structured linguistic data in other forms, such as a dictionary, into machine learning pipelines. We test a simple post-correction method from a pre-existing dictionary on Gitksan only, due to its unique combination of low resource and easily obtainable dictionary in machine readable form. We use the dictionary compiled by Forbes et al. (2021), without consulting the morphological analyzers that they also provided. At inference time, if a morpheme is unseen during training, we search for the exact form in the dictionary. We also expand the search to all sub se-

[1] https://en.wikipedia.org/wiki/Lezgin_alphabets

quences of morphemes within the enclosing word, plus the previous whole word in cases where a particle is included in the dictionary form. The first matched definition is used as the gloss and if none of the search yields an exact match, we fall back to the model prediction. We only apply this method to the token classification models because the alignment between morphemes and glosses is directly established, whereas the seq2seq models do not guarantee that the number of glosses matches the number of morphemes.

5 Results and Discussion

Tables 1 and 2 show our systems' performance (as well as the original baseline provided by the organizers) on the test data with respect to word- and morpheme-level micro-averaged accuracy, respectively. Overall, the token classification model trained first on the artificially generated augmented data perform the best, with the model trained on the shared task data only not far behind. Meanwhile, ByT5 models perform worse, with the model finetuned first on ODIN trailing our best model by a few percentage points, while the model finetuned first on augmented data performs worse than the baseline.

It is interesting that ByT5 performs worse than token classification models, even when it is finetuned on ODIN, which effectively doubles the amout of annotated data. The gaps of performance are uneven across the languages, and we do observe some language and training data specific issues that could contribute to the deficiency of the ByT5 model. For Gitksan, where only a very small amount of data is available, the ByT5 model suffers from hallucination, generating more glosses than it should. It also has the opposite problem for Lezgi, when sometimes fewer glosses than expected are generated for long sequences of inputs. For Uspanteko, we observe that some morphemes are glossed with three question marks (???), perhaps to denote uncertainty or unknown morphemes, and we think this might have caused the model to falsely generalize the symbols to other contexts like code switching. In general, ByT5 models are worse at alignment because that is not enforced between segmented morphemes and glosses.

5.1 Data Augmentation

Overall, we find data augmentation to be useful. With artificially generated data, we see the effects are perhaps greatest for the mid-resource languages (ddo, lez, ntu, nyb, usp), while the highest and lowest resourced languages did not receive much benefit from pretraining on the artificial data. We think this is perhaps because there is a "sweet spot" with respect to the amount of data that is required to train a model. If there is enough data already, in the case of Arapaho, then the noisiness of artificial data would outweigh the benefit of training on them. On the other end of the scale, Gitksan perhaps needs more synthetic data for data augmentation to yield meaningful improvements.

For ByT5 models, artificially generated data seem to have the opposite effect, where performance is significantly degraded. A speculation for this effect is the fact the pretrained model is more sensitive to the overall semantics of the input sentences, and since the artificially generated sentences could be nonsensical, they could be much noisier than we expect. On the other hand, pretraining on ODIN yields improvements for the majority of the languages[2]. This is encouraging since we did not perform much preprocessing for ODIN, and there is definitely still room to make the data cleaner and more internally consistent, which in turn should result in a better model.

5.2 Choice of Hyperparameters

We find the choice of hyperparameters of the token classification models to be necessarily language and dataset specific. Gitksan in particular needs special attention, where the number of training epochs need to be greatly increased for the very low data size. We find that the token classification model needs at least a few thousand steps to converge, and controlling for the number of minimum training steps makes more sense in low data settings. After the submission deadline has been concluded, we ran more experiments and discovered our Lezgi and Natugu models are under-trained. We do not include those latest experiments in this paper, but we believe our token classification models have the potential to perform better with more hyperparameter tuning.

5.3 In- Versus Out-of-Vocabulary Errors

One dimension of error analysis we investigated was what proportion of our systems' errors come

[2]Tsez is the only language that appeared in ODIN (68 sentences). We did not remove it from the corpus but this should have little influence on the performance because the size of the dataset is very small.

Model	arp	ddo	git	lez	ntu	nyb	usp	AVG
xlmr-base	**85.87**	73.77	27.86 / **34.11**[a]	74.15	**82.99**	80.61	73.47	72.14
xlmr-aug	82.92	**80.07**	24.74 / 31.25	**77.77**	78.72	**85.53**	**77.51**	**73.39**
byt5-base	78.86	80.32	14.84	60.72[b]	76.67	76.73	77.21	66.48
byt5-aug	73.27	62.37	4.17	38.60	55.11	69.25	70.85	53.38
byt5-odin	80.56	82.79	20.57	63.77	77.97	82.59	75.72	69.14
baseline	85.44	75.71	16.41	34.54	41.08	84.30	76.55	59.14

[a]We report before / after dictionary based post-correction for Gitksan.
[b]We trained this model without romanizing Lezgi.

Table 1: Word-level accuracy of our submitted systems. Best performance per language in the table is **bolded**. The XLMR baseline is the highest Arapaho accuracy reported out of all shared task submissions.

Model	arp	ddo	git	lez	ntu	nyb	usp	AVG
xlmr-base	**91.36**	84.35	47.47 / **52.82**	80.17	**88.35**	85.84	80.08	80.42
xlmr-aug	89.34	**88.15**	46.89 / 52.39	**82.36**	85.53	**89.49**	**83.08**	**81.48**
byt5-base	78.82	75.77	12.59	44.10	62.40	78.97	74.25	60.99
byt5-aug	72.10	57.93	2.60	26.24	35.62	70.01	67.73	47.46
byt5-odin	80.81	78.24	12.74	50.00	63.39	85.30	73.25	63.39
baseline	91.11	85.34	25.33	51.82	49.03	88.71	82.48	67.69

Table 2: Morpheme-level accuracy of our submitted systems. Best performance per language in the table is **bolded**. The XLMR baseline with artificial pretraining and dictionary post-correction is the highest Gitksan accuracy reported out of all shared task submissions.

from morphemes or words that are either in or out of the training data vocabulary. We count a morpheme or word as in-vocabulary if the surface form and its corresponding gloss co-occur in the provided training data (not including the development data, as our models are only trained on the train set). Note that there is a much larger proportion of OOV words as opposed to morphemes due to the fact that an unseen word can be composed of different combinations of seen morphemes.

Table 3 shows the proportion of morphemes and words that are out-of-vocab (OOV) within the test set. While nearly all the languages have less than 10% of their morphemes classified as OOV, Gitksan notably has a relatively large portion of OOV test data, with $\approx 45\%$ of morphemes and $\approx 78\%$ of words being OOV.

Tables 4 and 5 show our models' performances on in- verses out-of-vocab tokens at the morpheme and word levels, respectively. While we would intuitively expect that word-level OOV accuracy be about the same or worse than morpheme-level OOV accuracy, this is not the case due to the fact that a large portion of out-of-vocab words are formed with in-vocab morphemes. For most languages, with the exception of Gitksan, there appears to be a trade-off between better in-vocab morpheme performance with XLMR and performance out-of-vocab with ByT5.

	arp	ddo	git	lez	ntu	nyb	usp
Morph	4.3	0.9	45.0	5.6	3.4	1.9	7.0
Word	24.2	15.5	78.1	16.9	21.4	8.4	20.0

Table 3: Percentage of morphemes and words that are OOV within the test set.

6 Related Work

There have been a variety of approaches to the problem of (semi-) automatically generating interlinear glosses. Baldridge and Palmer (2009) investigate the efficacy of active learning for the task of interlinear glossing, using annotation time required by expert and non-expert annotators as their metric. The system they use to generate gloss label suggestions is a standard maximum entropy classifier.

Model	arp	ddo	git	lez	ntu	nyb	usp
xlmr-base	95.20	85.12	82.89	84.79	90.87	87.46	86.05
	4.97	0.00	16.08	2.60	14.52	0.00	0.82
xlmr-aug	92.98	88.94	84.74	87.10	87.88	91.17	89.31
	7.49	0.00	12.86	2.60	19.35	0.00	0.41
byt5-aug	74.76	58.24	3.42	40.27	36.54	71.27	70.56
	12.31	24.10	1.61	23.54	9.68	3.23	30.20
byt5-odin	83.47	78.55	18.42	62.90	64.38	86.85	75.23
	21.14	43.37	5.79	47.52	35.48	3.23	46.94

Table 4: Morpheme-level accuracy over all tokens of our submitted systems, split by in- versus out-of-vocab. Cells highlighted in gray indicate OOV accuracy.

Model	arp	ddo	git	lez	ntu	nyb	usp
xlmr-base	95.93	78.18	95.23	84.24	93.14	85.85	86.27
	54.44	49.79	17.00	24.67	45.65	23.60	22.41
xlmr-aug	93.72	83.85	94.05	87.64	89.24	90.81	91.11
	49.17	59.51	13.67	29.33	40.00	23.24	28.09
byt5-aug	87.22	68.69	10.71	46.06	65.13	74.59	81.44
	29.69	28.04	2.33	2.00	18.26	11.24	28.63
byt5-odin	91.93	87.66	63.10	73.78	85.93	87.60	83.46
	45.07	56.36	8.67	14.67	48.70	28.09	44.81

Table 5: Word-level accuracy of our submitted systems, split by in- versus out-of-vocab. Cells highlighted in gray indicate OOV accuracy.

A rule-based approach by Snoek et al. (2014) utilizes an FST to generate glosses for Plains Cree, focusing on nouns. Samardžić et al. (2015) view the task of glossing segmented text as a two-step process, first treating it as a standard POS tagging task and then adding lexical glosses from a dictionary. They demonstrate this method on a Chintang corpus of about 1.2 million words.

A number of other works focusing on interlinear glossing utilize conditional random field (CRF) models. Moeller and Hulden (2018) test three different models on a very small Lezgi dataset (< 3000 words): a CRF (that outputs BIO labels with the corresponding gloss per character in the input), a segmentation and labelling pipeline that utilizes a CRF (for BIO labels) and SVM (for gloss labels), and an LSTM seq2seq model. They find that the CRF that jointly produces the BIO labels and tags produced the best results. McMillan-Major (2020) utilizes translations in their training data by creating two CRF models, one that predicts gloss from the segmented input and another than predicts from the translation, and then uses heuristics to determine which model to select from for each morpheme. Barriga Martínez et al. (2021) used a CRF model to achieve $> 90\%$ accuracy for glossing Otomi and find that it works better than an RNN, which is computationally more expensive.

Other works, including our systems, have turned to neural methods. Kondratyuk (2019) leverages pretrained multilingual BERT to encode input sentences, then apply additional word-level and character-level LSTM layers before jointly decoding lemmas and morphology tags using simple sequence tagging layers. Furthermore, they show that two-stage training by first training on all languages followed by training on the target language is more effective than training the system on the target language alone. An approach by Zhao et al. (2020), like McMillan-Major (2020), makes use of translations available in parallel corpora, but do so by using a multi-source transformer model. They also incorporate length control and alignment during inference to enhance their model, and test their

system on Arapaho, Tsez, and Lezgi.

Past SIGMORPHON shared task have inspired the use of data augmentation to improve performance in low resource settings. Silfverberg et al. (2017) generated additional examples by randomly replacing the characters in the stem with other invocabulary characters. Anastasopoulos and Neubig (2019) employed a similar method, while keeping the lengths of generated stems the same as the original. However, these methods were designed for reinflection, where stem is not glossed, and our method that does not generate new stems is presumably more fitting for the present task. Additionally, Bergmanis et al. (2017) used a more sophisticated method that automatically finds orthographic patterns from an unlabeled word list to create pseudo annotations. This method can be modified for glossing, but it requires unlabeled word lists which is not guaranteed to be available for all low resource languages.

7 Conclusion

In our shared task submission, we explore data augmentation methods and modeling strategies for the task of interlinear glossing in seven low-resource languages. Our best performing models are token classification models using XLMR. We demonstrate that pretraining on artificial data with XLMR is an effective technique for the mid-resource test languages. Additionally, in our error analysis we find that we may have actually undertrained our token classification models, and thus our systems may have the potential to perform better with additional hyperparameter tuning. While our ByT5 models did not perform as well as our other systems, we show that pretraining on ODIN data is effective, despite this data being very noisy. Finally, we also demonstrate improvements by utilizing a dictionary to post-correct model outputs for Gitksan.

Acknowledgements

This work was supported by NSF CISE RI grant number 2211951, From Acoustic Signal to Morphosyntactic Analysis in one End-to-End Neural System.

References

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 984–996, Hong Kong, China. Association for Computational Linguistics.

Jason Baldridge and Alexis Palmer. 2009. How well does active learning *actually* work? Time-based evaluation of cost-reduction strategies for language documentation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 296–305, Singapore. Association for Computational Linguistics.

Diego Barriga Martínez, Victor Mijangos, and Ximena Gutierrez-Vasques. 2021. Automatic interlinear glossing for Otomi language. In *Proceedings of the First Workshop on Natural Language Processing for Indigenous Languages of the Americas*, pages 34–43, Online. Association for Computational Linguistics.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 31–39, Vancouver. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Clarissa Forbes, Garrett Nicolai, and Miikka Silfverberg. 2021. An FST morphological analyzer for the gitksan language. In *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 188–197, Online. Association for Computational Linguistics.

Michael Ginn, Sarah Moeller, Alexis Palmer, Anna Stacey, Garrett Nicolai, Mans Hulden, and Miikka Silfverberg. 2023. Findings of the SIGMORPHON 2023 shared task on interlinear glossing. In *Proceedings of the 20th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. Association for Computational Linguistics.

Dan Kondratyuk. 2019. Cross-lingual lemmatization and morphology tagging with two-stage multilingual BERT fine-tuning. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 12–18, Florence, Italy. Association for Computational Linguistics.

William D. Lewis and Fei Xia. 2010. Developing ODIN: A Multilingual Repository of Annotated Language Data for Hundreds of the World's Languages. *Literary and Linguistic Computing*, 25(3):303–319.

Angelina McMillan-Major. 2020. Automating gloss generation in interlinear glossed text. *Proceedings of the Society for Computation in Linguistics*, 3(1):338–349.

Sarah Moeller and Mans Hulden. 2018. Automatic glossing in a low-resource setting for language documentation. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 84–93.

David R. Mortensen, Siddharth Dalmia, and Patrick Littell. 2018. Epitran: Precision G2P for many languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

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

Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Tutorials*, pages 15–18.

Tanja Samardžić, Robert Schikowski, and Sabine Stoll. 2015. Automatic interlinear glossing as two-level sequence classification.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4596–4604. PMLR.

Miikka Silfverberg, Adam Wiemerslage, Ling Liu, and Lingshuang Jack Mao. 2017. Data augmentation for morphological reinflection. In *Proceedings of the CoNLL SIGMORPHON 2017 Shared Task: Universal Morphological Reinflection*, pages 90–99, Vancouver. Association for Computational Linguistics.

Conor Snoek, Dorothy Thunder, Kaidi Lõo, Antti Arppe, Jordan Lachler, Sjur Moshagen, and Trond Trosterud. 2014. Modeling the noun morphology of Plains Cree. In *Proceedings of the 2014 Workshop on the Use of Computational Methods in the Study of Endangered Languages*, pages 34–42, Baltimore, Maryland, USA. Association for Computational Linguistics.

Fei Xia, William Lewis, Michael Wayne Goodman, Joshua Crowgey, and Emily M. Bender. 2014. Enriching ODIN. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3151–3157, Reykjavik, Iceland. European Language Resources Association (ELRA).

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Xingyuan Zhao, Satoru Ozaki, Antonios Anastasopoulos, Graham Neubig, and Lori Levin. 2020. Automatic interlinear glossing for under-resourced languages leveraging translations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5397–5408, Barcelona, Spain (Online). International Committee on Computational Linguistics.

A Hyperparameter Settings

We use Adafactor (Shazeer and Stern, 2018) as the optimizer across all experiments, with the default scheduler from Hugging Face Transformers, a batch size of 32 for XLMR based models and a batch size of 4 with a gradient accumulation step of 8 for ByT5 based models. We train the token classification models for 40 epochs except for Arapaho, on which we train 20 epochs, and Gitksan, on which we train 2,000 steps. We train the ByT5 based models for 20 epochs on all of the data mixed together.