How low is too low? A monolingual take on lemmatisation in Indian languages

Kumar Saunack*, **Kumar Saurav***, **Pushpak Bhattacharyya**
IIT Bombay
{krsaunack, krsrv, pb}@cse.iitb.ac.in

Abstract

Lemmatization aims to reduce the sparse data problem by relating the inflected forms of a word to its dictionary form. Most prior work on ML based lemmatization has focused on high resource languages, where data sets (word forms) are readily available. For languages which have no linguistic work available, especially on morphology or in languages where the computational realization of linguistic rules is complex and cumbersome, machine learning based lemmatizers are the way to go. In this paper, we devote our attention to lemmatisation for low resource, morphologically rich scheduled Indian languages using neural methods. Here, low resource means only a small number of word forms are available. We perform tests to analyse the variance in monolingual models' performance on varying the corpus size and contextual morphological tag data for training. We show that monolingual approaches with data augmentation can give competitive accuracy even in the low resource setting, which augurs well for NLP in low resource setting.

1 Introduction

Natural Language Processing (NLP) has seen remarkable growth in all its sub-areas like machine translation, summarization, question answering and so on. For all these tasks, though, morphemes remain the most basic form of information (Otter et al., 2020). Morpheme identification (lemma and affixes) can assist these very useful large applications by solving the data sparsity problem.

Good lemmatisers are invaluable tools for handling large vocabulary in morphologically rich languages and thereby boosting performance in downstream tasks, but techniques are limited by resource availability. This is a relevant point for Indian languages. For instance, as many as 197 Indian languages are in the UNESCO's Atlas of the "World's Languages in Danger, 2010". Even among the 22 scheduled languages of India, there is a wide disparity in resource availability, for example, for Konkani and Kashmiri (Rajan et al., 2020; Islam et al., 2018).

Techniques like Porter stemmer are indeed quick solutions, but they are suited only for alphabetic script languages, like English, and not abugida, like Bengali (Ali et al., 2017), or abjad, like Urdu (Kansal et al., 2012), script languages. Moreover, creating stemmers requires different language specific stemming algorithms. This requirement of language specific measures comes in the way of scaling the enterprise of creating stemmers for the hundreds and thousands of languages that exist in the world. One might think of ML for stemming- for example, training a neural net with stems and word forms; but almost none of the 22 scheduled Indian languages, which is just a subset of the numerous languages spoken and written in India, have resources sufficient for training deep models (Bhattacharyya et al., 2019). For a majority of Indian languages, the absence of dictionaries compounds the problem.

Most of the current approaches for morphological analysis use the idea of cross-lingual transfer learning from a higher resource language to the low resource language (McCarthy et al., 2019) of interest. We show that even monolingual models can consistently perform with high accuracy with even as little as 500 samples, without cross-lingual training of neural models and without structured information like dictionaries. We further demonstrate good performance in extremely low resource

[*] These authors contributed equally to this work

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4088–4094
June 6–11, 2021. ©2021 Association for Computational Linguistics

setting with as few as 100 training examples samples to train on and show a competitive performance against cross-lingual models in the same setting.

2 Related work

In Zeman et al. (2018), lemmatisation was performed for small treebanks exploiting the common annotation standard across all languages, and the same task was implicit in Nivre et al. (2017). Recently, there has been a shift to extremely low resource settings with the SIGMORPHON 2019 shared task (McCarthy et al., 2019) focusing on cross-lingual learning. However, their task focuses on the reverse direction: given a lemma and a set of morphological features, generate a target inflected form.

3 Models

A two-step attention process (Anastasopoulos and Neubig, 2019) similar to the SIGMORPHON 2019 morphological inflection task (McCarthy et al., 2019) has been adapted for the setup, which consists of four components: encoder for morphological tags, encoder character sequence, attention and a decoder.

The inputs to the model are inflected words and morphological tags, and we use self-attention single layer bidirectional LSTM without positional embeddings as encoders. At each time step, during decoding, two context vectors are created via two different attention matrices over the output from the encoding of inflected word and morphological tag.

At the decoder, we use a two-step process: first we create a tag-informed state by attending over tags using the output from the decoder at the previous time step. Second, we use this to attend over the source characters to produce the state vector for the decoder at that time step, which is used for producing the output character for that time step using a fully connected layer followed by a softmax.

We also add structural bias to the attention model that encourages Markov assumption over alignments, that is, if the i-th source character is aligned to the j-th target one, alignments from the $(i+1)$-th or ith to $(j+1)$-th character are preferred.

We refer the reader to Anastasopoulos and Neubig (2019) for more details and explanations about the two-step attention process and Cohn et al. (2016) for more details regarding structural bias.

4 Experiments

4.1 Data

From the SIGMORPHON 2019 shared task, we collect language data from the multilingual morphological inflection task for Bengali, Hindi, Kannada, Sanskrit, Telugu, and Urdu. Out of these, Telugu is the only one that does not have a large data set (inflected word forms). We use the same task categorization of high or low resource languages as SIGMORPHON. Each training sample is a triplet: `(inflected word, lemma, tag)`, where `tag` refers to the set of morphological features for the inflected word.

A detailed description of the dataset that we use for training is provided in Table 1.

Language	Total	High	Low
Bengali (bn)	3,394	3,394	100
Hindi (hi)	10,000	10,000	100
Kannada (kn)	3,506	3,506	100
Sanskrit (sa)	10,000	10,000	100
Telugu (te)	61	-	61
Urdu (ur)	10,000	10,000	100

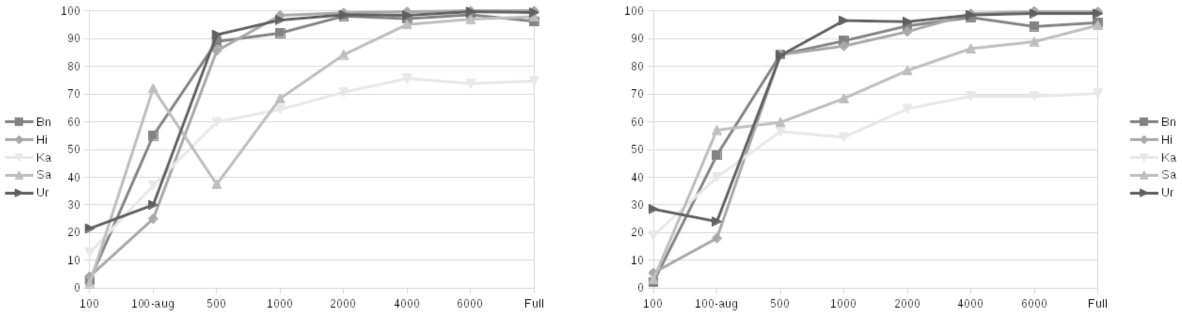
Table 1: Number of inflected-word lemma pairs available for each language. *Total* - original number of samples, *High* and *Low* - training dataset size in high and low resource settings.

We create the smaller data sets from the high-resource data sets using the sampling method based on probability distributions mentioned in Cotterell et al. (2018). During training for smaller data sets, we use augmentation from Cotterell et al. (2016). This particular augmentation method relies on substituting stems in a word with random sequences of characters while preserving its length.

We also annotate data sets with tag information to create multiple data sets for analysing the effects of data set size and the importance of tag information on the accuracy of the models.

4.2 Training

The model runs in two phases:



(a) Complete tag data

(b) Zero tag data

Figure 1: Accuracy when differing amounts of morphological data are used for training the models. Number of samples used in training are on X-axis (100-aug represents training dataset of 100 augmented to reach 10,000).

Warm-up Phase For each triple $(\mathbf{X}, \mathbf{Y}, \mathbf{T})$ in the original data, we create two new tuples $(\mathbf{X}, \mathbf{X}, [\mathrm{COPY}])$ and $(\mathbf{Y}, \mathbf{Y}, \mathbf{T})$ and train the model on the new tuples (Anastasopoulos and Neubig, 2019). This helps the model learn a monotonic alignment in the attention model, which is effective for character level transduction tasks (Wu and Cotterell, 2019) while avoiding any explicit modelling of such a structural bias. The training switches to the next phase when accuracy on the validation set exceeds 75%.
$(\mathbf{X}, \mathbf{Y}, \mathbf{T})$ triplet example for Hindi: (तू रहेगी, रहना, `V;V.PTCP;PST`). A Spanish example would be (`bailaba`, `bailar`, `V;V.PTCP;PRS`).

Main Phase The training tuple $(\mathbf{X}, \mathbf{Y}, \mathbf{T})$ is fed into the system, and the model is allowed to learn the distribution over the data. A cool down period is also used while training to improve the accuracy of the model. We also employ early stopping with a higher threshold than the cool down period so that the training stops when no further progress is possible.

Hyperparameters for our models are discussed in appendix A.1. We also release all our code online for reproducibility and further research. [*]

5 Results and Discussions

5.1 Variation with number of training word-pairs

We create three models for each training set size. They contain (1) no morphological features, (2) basic PoS tag data, and (3) all morphological features. We report accuracies over complete string matching for our experiments.

Figure 1 shows the graphs for accuracy versus data. When the complete set of morphological features is included in training, most languages achieve extremely high accuracy (at least 95%, except for Kannada), even when data set sizes are as small as 1000. When the data set size is 500, the accuracy drop to the range 80-90% but are still competitive wrt rule-based lemmatisers across languages (Bhattacharyya et al., 2014) like Sanskrit(Raulji and Saini, 2019), Hindi(Paul et al., 2013), Bengali(Shakib et al., 2019), Urdu(Gupta et al., 2015) and Kannada(Prathibha and Padma, 2015). However, the performance drops drastically when the data set size is reduced to 100. Performance on the augmented data sets shows a marked increase in accuracy over the unaugmented 100 training samples, but is still below the performance of models trained on 500 samples.

Telugu is not included in Figure 1 due to the lack of training samples. We train only one model over the available 61 samples (augmented to 10,000). The model achieves an accuracy of 80% on the SIGMORPHON Task 1 test set for Telugu.

5.2 Variation with morphological information

Comparing Figure 1(a) and 1(b), we see that tag data does not provide substantial additional information to the model when the data set size exceeds 2000, barring the case for San-

[*] `https://github.com/krsrv/lemmatisation`

	2000		1000		500		100		100-aug	
	No tag	PoS	No tag	PoS	No tag	PoS	No tag	PoS	No tag	PoS
bn	-3.60	0.00	-5.72	-2.74	-3.49	1.89	12.77	50.00	2.13	17.02
hi	2.44	9.97	-8.97	2.61	-6.65	-5.10	-5.17	-27.59	-30.77	-3.85
kn	-4.70	4.04	-18.15	-3.08	-7.47	-1.87	97.60	33.33	-4.76	-11.90
sa	-11.30	-4.86	-5.52	-5.52	-4.32	-40.16	40.91	-22.73	-9.52	14.29
ur	-2.93	-0.40	-1.73	-1.53	-8.41	-0.22	141.53	82.20	-36.84	-21.05

Table 2: Each column represents the percentage change in accuracy compared to the accuracy when all morph tags were used. *No tag* - no morph tag, *PoS* - basic PoS tag data as inputs. The topmost row represents the number of samples.

skrit. At 500, there is a spike in accuracy for Sanskrit which is probably explained by the fact that Sanskrit is a morphologically and semantically systematic language with very few ambiguities (evident from its linguistic and grammar text Aṣṭādhyāyī by Pāṇini), and thus is the language with highest responsiveness to augmentation with tag data. Below 4000, the morphological tag data substantially improves the accuracy. Sanksrit and Kannada both show worse results compared to other languages, which is likely due to the complex inflection patterns in both languages.

The gains from including tag information are better visualised in Table 2. A negative value in the table indicates that the model's performance decreases in absence of tag data. In general, we see that full-tag informed models perform the best, followed by basic PoS tag informed models and finally models without tag information.

The table also shows that the importance of tag data increases considerably with decrease in the training set size. However, an anomaly occurs with 100 training samples, when the absence of tag information improves the performance. A possible explanation is that the number of training samples is too low and the model is not able to learn what to focus on effectively. This anomaly disappears when we augment the data before training the model.

Note that achieving 100% accuracy on lemmatization without any tag information is not possible with *any* data set size. Some words can have multiple lemmas and require context for disambiguation: की (kee) can map to either करना (karana) or का (kaa) depending on whether it is used as a postposition or a verb.

	bn	hi	kn	sa	ur	Avg cross	mono
bn	-	60	59	57	59	58.8	55
hi	45	-	45	45	45	45	26
kn	52	53	-	44	48	49.3	42
sa	70	68	74	-	70	70.5	72
ur	24	23	20	10	-	19.3	38

Table 3: Accuracy of the cross-lingual model on different language pairs. Columns: high resource languages, Rows: low resource languages. (Accuracy is measured via a complete string match.)

5.3 Comparison with cross-lingual models

We also train cross models using the same method as monolingual training and incorporate the training procedure described by Artetxe et al. (2020) (the hyperparameters are listed in appendix A.2). We simulate a low resource language by choosing 100 samples at random and use all the other languages as high resource languages. Macro averaged accuracy for a simulated low resource language shows that monolingual models give comparable accuracies when compared to cross-lingual models, with the exception of Hindi. Performance of Sanskrit and Urdu, especially Urdu, seem to be better when the mono-lingual models are used.

The complete list of accuracies for the cross-lingual models are listed in Table 3. The macro-averaged difference between the cross-lingual and monolingual model is -2 in the cross-lingual models' favor.

6 Conclusion

We have given a methodology for lemmatization of low resource (i.e., availability of small number of word forms) in this paper. For most languages, a monolingual model trained on approximately 1000 training samples gives competitive accuracy, while training on 500 samples gives results at par with rule-based linguistic systems. For extremely-low resource settings as well, monolingual models perform well with the help of data augmentation. Even in these scenarios, monolingual models can give competitive results compared to cross-lingual models, a result that is supported by research in other tasks such as morphological inflection (Anastasopoulos and Neubig, 2019).

Additionally, in the low resource setting, additional features are an important source of information. Even PoS tags benefit the training process.

6.1 Areas of improvement

The model currently does not exploit any linguistic knowledge available to improve its performance. Incorporating morphological rules or using bilingual knowledge to create transfer models could grant accuracy gains (Gebreselassie et al., 2020; Faruqui et al., 2015). Moreover, transformers have been shown to improve performance on character level tasks which would be applicable method here (Wu et al., 2020). Another potential area of improvement could be the usage of different data hallucination techniques like in Shcherbakov et al. (2016), which uses phonetics instead of relying on characters for predictions.

7 Ethical Considerations

The work in this paper can be useful for expanding the power of language understanding to ethnic/local languages. This can consequently bring these low-resource language domains within the umbrella of widespread NLP applications in edge computing devices. By focusing on low-resource domains, we understand how lightweight models fare in these settings, thereby leading to potential trimming down of model sizes, training time, compute costs etc., which is a significant step towards maintaining energy and carbon costs.

Such developments also spur the progress of languages and the civilisations associated with them by bringing them into the advanced technological manifolds, and thereby bring more equitable distribution of technology and quality of life across the globe.

References

Mubashir Ali, Shehzad Khalid, and Muhammad Haseeb Aslam. 2017. Pattern based comprehensive urdu stemmer and short text classification. *IEEE Access*, 6:7374–7389.

Antonios Anastasopoulos and Graham Neubig. 2019. Pushing the limits of low-resource morphological inflection. *arXiv preprint arXiv:1908.05838*.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Pushpak Bhattacharyya, Ankit Bahuguna, Lavita Talukdar, and Bornali Phukan. 2014. Facilitating multi-lingual sense annotation: Human mediated lemmatizer. In *Proceedings of the Seventh Global Wordnet Conference*, pages 224–231.

Pushpak Bhattacharyya, Hema Murthy, Surangika Ranathunga, and Ranjiva Munasinghe. 2019. Indic language computing. *Communications of the ACM*, 62(11):70–75.

Trevor Cohn, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Arya D. McCarthy, Katharina Kann, Sabrina J. Mielke, Garrett Nicolai, Miikka Silfverberg, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 1–27, Brussels. Association for Computational Linguistics.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The sigmorphon 2016 shared task—morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2015. Morphological inflection generation using character sequence to sequence learning. *arXiv preprint arXiv:1512.06110*.

Tewodros Gebreselassie, Amanuel Mersha, and Michael Gasser. 2020. A translation-based approach to morphology learning for low resource languages. In *Proceedings of the The Fourth Widening Natural Language Processing Workshop*, pages 36–40.

Vaishali Gupta, Nisheeth Joshi, and Iti Mathur. 2015. Design & development of rule based inflectional and derivational urdu stemmer 'usal'. In *2015 International conference on futuristic trends on computational analysis and knowledge management (ABLAZE)*, pages 7–12. IEEE.

Saiful Islam, Abhijit Paul, Bipul Shyam Purkayastha, and Ismail Hussain. 2018. Construction of english-bodo parallel text corpus for statistical machine translation. *International Journal on Natural Language Computing (IJNLC) Vol*, 7.

Rohit Kansal, Vishal Goyal, and Gurpreet Singh Lehal. 2012. Rule based urdu stemmer. In *Proceedings of COLING 2012: Demonstration Papers*, pages 267–276.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sabrina J. Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Morphological analysis in context and cross-lingual transfer for inflection. In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 229–244, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre, Lars Ahrenberg ˇZeljko Agic, et al. 2017. Universal dependencies 2.0–conll 2017 shared task development and test data. *LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University*.

Daniel W Otter, Julian R Medina, and Jugal K Kalita. 2020. A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*.

Snigdha Paul, Mini Tandon, Nisheeth Joshi, and Iti Mathur. 2013. Design of a rule based hindi lemmatizer. In *Proceedings of Third International Workshop on Artificial Intelligence, Soft Computing and Applications, Chennai, India*, pages 67–74. Citeseer.

RJ Prathibha and MC Padma. 2015. Design of rule based lemmatizer for kannada inflectional words. In *2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, pages 264–269. IEEE.

Annie Rajan, Ambuja Salgaonkar, and Ramprasad Joshi. 2020. A survey of konkani nlp resources. *Computer Science Review*, 38:100299.

Jaideepsinh K Raulji and Jatinderkumar R Saini. 2019. Sanskrit lemmatizer for improvisation of morphological analyzer. *Journal of Statistics and Management Systems*, 22(4):613–625.

MD Shahidul Salim Shakib, Tanim Ahmed, and KM Azharul Hasan. 2019. Designing a bangla stemmer using rule based approach. In *2019 International Conference on Bangla Speech and Language Processing (ICBSLP)*, pages 1–4. IEEE.

Andrei Shcherbakov, Ekaterina Vylomova, and Nick Thieberger. 2016. Phonotactic modeling of extremely low resource languages. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 84–93.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1530–1537, Florence, Italy. Association for Computational Linguistics.

Shijie Wu, Ryan Cotterell, and Mans Hulden. 2020. Applying the transformer to character-level transduction. *arXiv preprint arXiv:2005.10213*.

Daniel Zeman, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. Conll 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 1–21.

A Appendix

All our models were trained on a single 12 GB Nvidia GeForceGTX TitanXGPU and finished training within an hour.

A.1 Hyperparameters

We use the Adam optimiser with the default parameters except for learning rate. The training time for each model was between 1 to 3 hours. We list out the hyperparameters used by use during training:

- Batch size: 10

- Training epochs: 10
- Activation function: Squish
- Learning rate: 10e-3

All the hyperparameters were tuned using the validation set over all languages over a uniform distribution and the best models were selected based on accuracy.

A.2 Cross-lingual models

The cross-lingual method that we use corresponds to the method described by Artexte et al. (2019) and so there are 4 phases of training. We list out the hyperparameters as comma separated values:

- Batch size: 10
- Training epochs: 10,10,10,10
- Activation function: Squish
- Learning rate: 10e-3,10e-3,10e-3,10e-3

The 4 phases refer to the following:

- P1 : training on high resource language, with same output and input (copying phase for high resource language)
- P2 : training on low resource language, with same output and input (copying phase for low resource language)
- P3 : training on high resource language, with expected input (inflected word + tag) and output (lemma)
- P4 : training on low resource language, with expected input (inflected word + tag) and output (lemma)