On Efficiently Acquiring Annotations for Multilingual Models

Joel Ruben Antony Moniz[*], **Barun Patra**[*]
{jramoniz, barunpatra95}@gmail.com

Matthew R. Gormley
Carnegie Mellon University
mgormley@cs.cmu.edu

Abstract

When tasked with supporting multiple languages for a given problem, two approaches have arisen: training a model for each language with the annotation budget divided equally among them, and training on a high-resource language followed by zero-shot transfer to the remaining languages. In this work, we show that the strategy of joint learning across multiple languages using a single model performs substantially better than the aforementioned alternatives. We also demonstrate that active learning provides additional, complementary benefits. We show that this simple approach enables the model to be data efficient by allowing it to arbitrate its annotation budget to query languages it is less certain on. We illustrate the effectiveness of our proposed method on a diverse set of tasks: a classification task with 4 languages, a sequence tagging task with 4 languages and a dependency parsing task with 5 languages. Our proposed method, whilst simple, substantially outperforms the other viable alternatives for building a model in a multilingual setting under constrained budgets.

1 Introduction

While neural networks have become the de-facto method of tackling NLP tasks, they often require a lot of annotated data to do so. This task of data annotation is especially challenging while building systems aimed at serving numerous languages. Motivated by this, in this paper, we tackle the following problem:

Given the requirement of building systems for an NLP task in a multilingual setting with a fixed annotation budget, how can we efficiently acquire annotations to perform the task well across multiple languages?

The traditional approach to this problem has been building a separate model to serve each language. In this scenario, the annotation budget is split equally for all languages, and a model is trained for each one separately. Recently, another direction that has gained popularity has been leveraging multilingual pre-trained language models (MPLMs) which inherently map multiple languages to a common embedding space (Devlin et al., 2019; Conneau et al., 2020). The popular method for leveraging these models has been leveraging their zero-shot transfer ability: training on an English-only corpus for the task, and then using the models zero-shot for the other languages.

Another orthogonal line of work aimed at building models under a constrained budget has been active learning (AL) (Shen et al., 2018; Ein-Dor et al., 2020). While this has shown to improve annotation efficiency, the predominant approach has been to train one model per language, using the (language specific) model for AL (Shen et al., 2018; Erdmann et al., 2019).

In this work, we show that a single MPLM trained on all languages simultaneously performs much better than training independent models for specific languages for a fixed total annotation budget. Further, while the benefits of using AL in conjunction with MPLMs has been studied for a monolingual setup (Ein-Dor et al., 2020), we show that AL also yields benefits in the multilingual setup. Concretely, we show that an AL acquisition on a single language helps improve zero-shot performance on *all other* languages, regardless of the language of the seed data. Furthermore, we show that AL also yields benefits for our proposed single model scenario. We demonstrate that our results are consistent on 3 different tasks across multiple languages: classification, sequence tagging and dependency parsing. Our approach removes the requirement of maintaining n different models, and uses $1/n^{th}$ the parameters than when independent models are trained. Our analysis reveals that the model arbitrates between different languages based on its performance to form a multilingual curricu-

[*]Equal Contribution

lum.

We release our code at https://github.com/codedecde/SMAL.

2 Related Work

Effective utilization of annotation budgets has been the area of focus for numerous active learning works, showing improvements for different tasks like POS tagging (Ringger et al., 2007), sentiment analysis (Karlos et al., 2012; Li et al., 2013; Brew et al., 2010; Ju and Li, 2012), syntactic parsing (Duong et al., 2018), and named entity recognition (Settles and Craven, 2008; Shen et al., 2018). The focus of most of these works, however, has been on learning for a single language (often English). Prior work on AL that uses a multilingual setup or cross-lingual information sharing and that goes beyond training a separate model for each language has thus been limited. The closest work where multiple languages influence each other's acquisition is that of Qian et al. (2014); however, they still train a separate model for each language.

For transfer to multiple languages, recent advances in building MPLMs (Devlin et al., 2019; Conneau et al., 2020; Liu et al., 2020; Xue et al., 2020) have been extremely effective, especially in zero-shot transfer (Pires et al., 2019; Liu et al., 2020). Ein-Dor et al. (2020) studied the data-effectiveness of these models when used in conjunction with AL, but, as with other AL work, with a single language focus. Finally, Lauscher et al. (2020) studied the effectiveness of the zero-shot setup, showing that adding a few examples to a model trained on English improves performance over zero-shot transfer. However, this assumes the availability of a full English task-specific corpus.

3 Methodology

3.1 Task Specific Models

We use the multilingual-BERT-cased model (mBERT) as the base model for all the tasks. We use the standard training methodology for the tasks: For **classification**, we use a single layer over the [CLS] embedding. For **sequence tagging**, we use a single layer for each word to predict its tag. For **dependency parsing**, we follow Kondratyuk and Straka (2019) and use mBERT embeddings with the graph-based bi-affine attention parser (Dozat and Manning, 2017); refer Appendix A for details.

3.2 Budget Allocation Settings

To understand data acquisition in a multilingual setting, we consider multilingual datasets in the 3 tasks. For each task t, let $\mathcal{L}$ be the set of languages ($n = |\mathcal{L}|$). We then define s_t to be the seed size, b_t to be the total annotation budget and v_t to be total number of annotated validation examples available to t. We compare our proposed Single Model Acquisition (SMA) setup to two baseline settings– Monolingual Acquisition (MonoA) and Multi Model Acquisition (MMA):

MonoA In this setting, the seed data as well as the validation data (s_t, v_t) is acquired from a single language. Further, the entire annotation budget (b_t) is assigned to the same language. We evaluate the test data performance on that language and on the other $n - 1$ languages in a zero-shot setting.

MMA For this setting, we train n individual models, one for each language. Each model starts with a seed of s_t/n, a validation set of v_t/n, and is assigned an acquisition budget of b_t/n. At test time, we evaluate the performance of the model on the language it was trained with.

SMA For this setting, we consider a single model for which both training and acquisition is done on all n languages simultaneously. The seed data and the validation set comprises of a random subset drawn from data corresponding to all languages. The whole of s_t, b_t and v_t are thus assigned to this single model. We compute the performance on the test data of each of the languages.

3.3 Active Learning Acquisition Strategies

The field of active AL tends not to reveal explicit winners—though there is a general consensus that AL does indeed outperform passive learning (Settles, 2009). Thus, we adopt the simplest confidence based strategies to demonstrate their efficacy for each task : Least Confidence (LC) for classification, Maximum Normalized Log Probability (MNLP) (Shen et al., 2018) for sequence tagging, and normalized log probability of decoded tree (NLPDT) (Li et al., 2016) for dependency parsing

Maximum Normalized Log Probability (MNLP) This strategy chooses instances for which the log probability of the model prediction, normalized by sequence length, is the lowest. This AL strategy has been shown to be extremely effective for NER

(Shen et al., 2018) and hence we adopt it in our setting.

Least Confidence (LC) This strategy chooses those instances for which the model confidence corresponding to the predicted class is the least. This acquisition strategy has been commonly applied in classification tasks, and although simple, has been consistently shown to often perform extremely well (Settles, 2009); consequently, we adopt it in our setting.

Normalized Log Probability of the Decoded Tree (NLPDT) This strategy selects the instances with the minimum log probability of the decoded tree generated d^* as generated by the Chu-Liu/Edmonds algorithm (refer A for additional details). Following (Li et al., 2016), we also normalize this score by the number of tokens N [1].

To the best of our knowledge, this is the first work to explore an AL-augmented single model for multiple languages.

4 Experiments

4.1 Dataset Details

Classification We consider Sentiment Analysis, using the Amazon Reviews dataset (Prettenhofer and Stein, 2010). The dataset consists of reviews and their binary sentiments for 4 languages: English (en), French (fr), Japanese (ja), German (de).

Sequence Tagging We choose Named Entity Recognition, and use the CoNLL02/03 datasets (Sang, 2002; Tjong Kim Sang and De Meulder, 2003) with 4 languages: English (en), Spanish (es), German (de) and Dutch (nl), and 4 named entities: Location, Person, Organization and Miscellaneous.

Dependency Parsing We use a subset of treebanks with 5 languages (English (en), Spanish (es), German (de), Dutch (nl), Japanese (ja)) from the full Universal Dependencies v2.3 corpus (Nivre et al., 2018); a total of 11 treebanks.

4.2 Experimental Settings

For each experiment, we run 4 training rounds: one training on initial seed data, followed by 3 acquisition rounds. We set s_t=b_t=v_t in all cases. For classification, we set s_t=300 sentences. For NER and Dependency Parsing, we use s_t=$\sim$10k and s_t=$\sim$17.5k tokens respectively (refer Appendix B). We report accuracy for classification, F1-Score for the NER, and unlabeled and labeled attachment scores (UAS and LAS) for dependency parsing.

For each task, we run the 3 settings (§3.2) across multiple languages. For each setting, we also train an AL model with a task-specific acquisition function (§3.3). In addition, we train both the SMA and MMA with all available data, i.e., we use all data to train one model for all languages and one model per language respectively. We report an average of 5 runs for each experiment. Refer Appendix C for hyperparameters and training details.

5 Results and Analysis

Model Performance Figure 1 shows the performance of NER on Spanish (refer Appendix G for the plots of all other languages and tasks). Although acquiring data independently per language (MMA) performs well, SMA outperforms MMA. Unsurprisingly, MonoA with es performs the best in the category, since it allocates its entire budget to acquiring es data; it thus forms an upper-bound of the model performance. However, SMA outperforms MonoA when its seed language and inference language differ. Finally, AL consistently provides gains over random acquisition.

To analyze the performance across all languages, we present the performance for each round of acquisition, aggregated across all languages for Classification (Figure 2) (refer Appendix G for Dependency Parsing and NER plots). Here, SMA consistently outperforms MMA for every round of acquisition because MMA suffers from a poorly utilized budget, potentially wasting annotation budget on languages where the task is easier. In contrast, SMA improves budget utilization while also benefiting from cross-lingual information. Finally, SMA, by virtue of performing well irrespective of language, consistently outperforms MonoA.

For a concise overview, we present the aggregate metrics across all rounds for each task in Table 1. We observe that SMA does much better compared to its counterparts; both with and without AL. We also observe these models to be extremely data efficient: with AL, a model with access to less than 5% of the data achieves a (relative) performance of around 88% accuracy (for classification), 95.5% F1-score (for NER) and 93.5% LAS (for depen-

[1] We also tried normalizing by N^2, as well as a globally normalized probability of d^* (probability of the tree over all possible valid trees, with the partition function computed using the Matrix Tree Theorem (Koo et al., 2007; Smith and Smith, 2007)), but found both to perform worse.

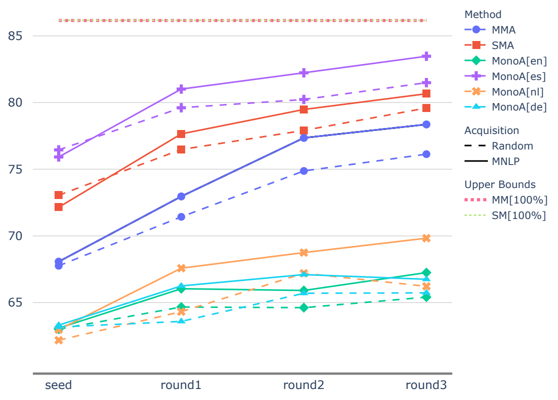


Figure 1: Performance across different rounds for one task (NER) and one language (es). Note that SMA ± AL out-performs MMA ± AL. It also out-performs all MonoA baselines except MonoA[es], which is the language specific upper bound. Here MNLP is the AL method adopted for NER.

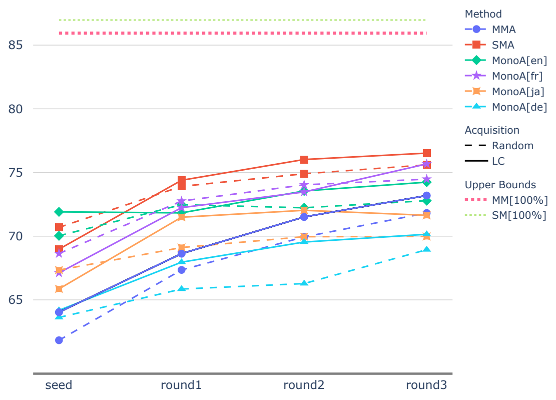


Figure 2: Performance aggregated across all languages for one task (classification) at every round of acquisition. As can be seen, SMA ± AL outperforms all other baselines. Note that SMA and MMA both out-perform MonoA. This is because MonoA does not perform as well when the language is different than that for which data was acquired. Here, LC is the AL method adopted for classification.

dency parsing) when compared to a model trained with all available data (see Table 2 for full data performance). Further, along with its superior performance, SMA also affords substantial parameter savings: requiring only a single model, compared to a number of models linear in n (thereby using $\frac{1}{n^{th}}$ parameters compared to MMA).

Dataset	Metric	AL	MMA	SMA
NER	Span-F1	(-)	75.1	79.1
		(+)	77.3	**80.5**
Classification	Accuracy	(-)	67.7	73.8
		(+)	69.3	**74.0**
Dependency Parsing	UAS	(-)	84.8	86.0
		(+)	84.5	**86.3**
	LAS	(-)	78.0	77.8
		(+)	77.8	**79.7**

Table 1: Average results across all rounds (5%, 10%, 15% and 20% data) and all languages. (+) and (-) indicate with and without AL respectively. Bold highlights best performance for a task.

MM Full vs SM Full To analyze how effectively a single model performs on the languages in question despite using $1/n^{th}$ the parameters, we train a single model on all data and compare it with n language-specific models, where each of the n models has the same number of parameters as the single model; this also serves as an upper-bound for our AL experiments. Table 2 shows that having a single model does not adversely impact performance. A more detailed discussion is present in Appendix D.

Dataset	Metric	MM Full	SM Full
NER	Span-F1	**87.4**	87.2
Classification	Accuracy	86.0	**87.0**
Dependency Parsing	UAS	**91.3**	**91.3**
	LAS	**87.1**	**87.1**

Table 2: Performance with all data for both SM and MM. Here, SM is a single model trained on all languages, while MM represents average performance over all languages of one model per language. The comparable performance indicates that models have enough capacity to represent languages in consideration.

The effectiveness of AL in MonoA We consistently observe AL in the source language improving performance across all languages, irrespective of whether inference is being run for the source language or zero-shot on a different target language, both for NER and classification (Table 3). We hypothesize that the model selects semantically difficult or ambiguous examples that generalize across languages by virtue of mBERT's shared embedding representation. To the best of our knowledge, this work is the first to demonstrate that AL can improve the data efficiency of both classification and NER in a zero-shot inference setup.

In the case of dependency parsing, we observe mixed results when the source and target languages differ. We hypothesize that this is because dependency parsing is a syntactic problem, making it more language specific, and zero-shot inference inherently harder. This is in contrast with both classification and NER, which are more semantic, making hard examples more generalizable across languages. Refer Appendix E for more details.

Dataset	Metric	AL	MonoA				
NER		Source	**en**	**es**	**nl**	**de**	
	Span-F1	(-)	71.3	64.3	68.8	68.8	
		(+)	**72.1**	64.3	70.8	70.3	
Classification		Source	**en**	**fr**	**ja**	**de**	
	Acc	(-)	71.9	72.5	69.1	66.2	
		(+)	**72.9**	72.1	70.3	68.0	
Dependency Parsing		Source	**en**	**es**	**nl**	**de**	**ja**
	UAS	(-)	76.4	72.9	73.9	72.9	44.3
		(+)	**76.9**	73.0	74.0	73.4	44.2
	LAS	(-)	67.2	62.3	62.8	61.8	31.8
		(+)	**67.5**	62.4	62.7	62.3	30.8

Table 3: Average results across all rounds (5%, 10%, 15% and 20% data) and all languages for MonoAL. Source indicates the language of data acquisition and for all other languages, inference is zero-shot. As can be seen, AL usually helps in the zero-shot setup.

What does SMA+AL acquire? One advantage of the SMA+AL setup is that the model can arbitrate between allocating its acquisition budget across different languages as training progresses. This is in contrast with training one model per language, where the models for languages with a high performance waste the overall budget by acquiring more than necessary, while models on languages where performance isn't as good under-acquire.

To investigate this, for each language and each round, we plot the relative difference (%) between cumulative tokens acquired by the SMA+AL model for that language, and the tokens acquired in expectation if acquisition was done randomly (refer Appendix F for more details). For each language, we also plot the relative performance difference of the language at that round compared to the performance when 100% data is available.

Figure 3 reveals the added benefit of SMA+AL for data acquisition for NER (refer Appendix F for other tasks): a single model can arbitrate between instances across languages automatically. The model initially acquires data from the high resource language (English). But as the training proceeds, the model favors acquiring data from

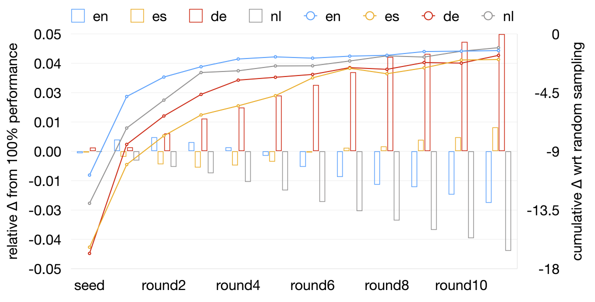


Figure 3: Acquisiton Curriculum for NER. The bars (left y-axis) represent the relative fraction of cumulative tokens acquired per language compared to random sampling. The lines (right y-axis) show the difference of performance of the language when compared to its 100% data performance (MM). Notice that the model tends to favor acquiring data from languages that underperform compared to their 100% counterpart (here, es and de). This in turn helps the model to arbitrate its acquisitions so as to achieve similar performance (relative to 100% performance) across all languages (indicated by the convergence of the line plots).

languages it is uncertain about (Spanish and German). This "multilingual curriculum" thus allows the model to be more effective in its use of the annotation budget. We find SMA+AL eventually achieves a similar relative difference from 100% data performance for all languages consistently across tasks as a consequence.

6 Conclusion

In this work, we consider the problem of efficiently building models that solve a task across multiple languages. We show that, contrary to traditional approaches, a single model arbitrating between multiple languages for data acquisition considerably improves performance in a constrained budget scenario, with AL providing additional benefits.

References

Anthony Brew, Derek Greene, and Pádraig Cunningham. 2010. Using crowdsourcing and active learning to track sentiment in online media. In *ECAI*.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14.

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. In *ACL*.

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

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *ICLR*.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2018. Active learning for deep semantic parsing. In *ACL*.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4).

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *EMNLP*.

Alexander Erdmann, David Joseph Wrisley, Benjamin Allen, Christopher Brown, Sophie Cohen-Bodénès, Micha Elsner, Yukun Feng, Brian Joseph, Béatrice Joyeux-Prunel, and Marie-Catherine de Marneffe. 2019. Practical, efficient, and customizable active learning for named entity recognition in the digital humanities. In *NAACL*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *NLP-OSS*.

Shengfeng Ju and Shoushan Li. 2012. Active learning on sentiment classification by selecting both words and documents. In *CLSW*.

Stamatis Karlos, Nikos Fazakis, Sotiris Kotsiantis, and Kyriakos Sgarbas. 2012. An empirical study of active learning for text classification. *ASSR*, 6(2).

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *EMNLP*.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *EMNLP*.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

Shoushan Li, Yunxia Xue, Zhongqing Wang, and Guodong Zhou. 2013. Active learning for cross-domain sentiment classification. In *IJCAI*.

Zhenghua Li, Min Zhang, Yue Zhang, Zhanyi Liu, Wenliang Chen, Hua Wu, and Haifeng Wang. 2016. Active learning for dependency parsing with partial annotation. In *ACL*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. In *TAACL*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Ahrenberg, et al. 2018. Universal dependencies 2.3. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *ACL*.

Peter Prettenhofer and Benno Stein. 2010. Cross-language text classification using structural correspondence learning. In *ACL*.

Longhua Qian, Haotian Hui, Ya'nan Hu, Guodong Zhou, and Qiaoming Zhu. 2014. Bilingual active learning for relation classification via pseudo parallel corpora. In *ACL*.

Eric Ringger, Peter McClanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, Kevin Seppi, and Deryle Lonsdale. 2007. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *LAW*.

Tjong Kim Sang. 2002. Ef: Introduction to the conll-2002 shared task. In *Proceedings of the 6th Conference on Natural Language Learning*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP*.

Yanyao Shen, Hyokun Yun, Zachary C. Lipton, Yakov Kronrod, and Animashree Anandkumar. 2018. Deep active learning for named entity recognition. In *ICLR*.

David A Smith and Noah A Smith. 2007. Probabilistic models of nonprojective dependency trees. In *EMNLP*.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, et al. 2020. Transformers: State-of-the-art natural language processing. In *EMNLP: System Demo*.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

A Task Specific Details

In this section, we elaborate on the task specific adaptations:

Classification: As is common practice, we use a single linear layer over [CLS] embeddings generated by the BERT model to generate logits for the classification task, and the model is trained to minimize the cross-entropy loss.

Sequence Tagging: We apply a linear layer to the word embeddings[2] generated by the BERT model to generate the tag logits, and the model is trained to minimize the negative log-likelihood of the observed tags.

Dependency Parsing: We use a graph-based biaffine attention parser introduced in (Dozat and Manning, 2017). Following (Kondratyuk and Straka, 2019), we use the output of the last BERT layer in place of the embeddings generated by the Bi-LSTM layers. These embeddings are then concatenated with the POS embeddings. A head feed-forward network and a child feed-forward network then generate embeddings for each head and dependant word of a dependency respectively. This is combined with a biaffine attention module to generate a probability distribution for each word to predict its head, as well as a bilinear layer to predict the label for each dependency relationship. Let $\tau_{(i)} = \{(h_{(i,j)}, d_{(i,j)}, l_{(i,j)} | h_{(i,j)} \rightharpoondown d_{(i,j)}$ with label $l_{(i,j)}\}$ be the i^{th} gold dependency tree in the dataset. The model is then trained to maximize the log probability of the gold tree as :

$$\max \sum_i \sum_j \log \left(\mathbb{P}(h_{(i,j)} | d_{(i,j)})\right) + \log \left(\mathbb{P}(l_{(i,j)} | h_{(i,j)} \rightharpoondown d_{(i,j)})\right) \tag{1}$$

During inference, the best dependency parse is generated by decoding with Chu-Liu/Edmonds algorithm (Chu, 1965; Edmonds, 1967).

For all the models mentioned above, all layers of mBERT are fine-tuned during training.

B Dataset statistics

We report the detailed dataset statistics in Table 4. Note that the seed was chosen to be roughly 5% of the size of the English training data, shown in the rightmost column of the table.

[2]Following (Devlin et al., 2019), for words generating multiple wordpieces, we use the embedding of the first wordpiece.

C Experimental Details

Hyperparameters All experiments performed in this paper are averaged over 5 runs. For each experiment, we perform an LR search over (1e-5, 2e-5, 3e-5, 4e-5 and 5e-5), and choose the best LR according to the performance on the appropriate validation (sub)set, as recommended in (Devlin et al., 2019). In all experiments, we set the batch size to 32 and use an Adam (Kingma and Ba, 2015) optimizer. Each round of training is run with a patience of 25 epochs, for at most 75 epochs in total.

Data Preprocessing To avoid out-of-memory issues on the GPU, we pre-process the data so that the examples in the train set of length larger than 175 and with larger than 256 word-pieces are filtered out for the NER. For classification, we simply truncate all instances at 256 word-pieces. We also de-duplicate the train set, to ensure that during all AL acquisition stages, no duplicates are selected at any point.

Code All code used in this work was implemented using Python, PyTorch and AllenNLP (Gardner et al., 2018), using pre-trained models released by HuggingFace (Wolf et al., 2020).

D SM Full vs MM Full Performance

Given that the SMA setup uses $1/n^{th}$ the number of parameters, an interesting question is whether fewer parameters leads to a loss in any expressive power for the single model, which might potentially lead to poorer performance (curse of multilinguality (Conneau et al., 2020)). To answer this question, we train a single model on all data and compare it with n language-specific models, where each of the n models has the same number of parameters as the single model.

From the 100% (rightmost) columns of Table 2, we find that having a single model does not adversely impact performance and these trends hold irrespective of whether all the languages in the task are etymologically close (as in NER) or distant (ja for classification and dependency parsing). This might not be the case when there are a large number of languages, however; investigating how well this observation scales with the number of languages would be an interesting line of future work.

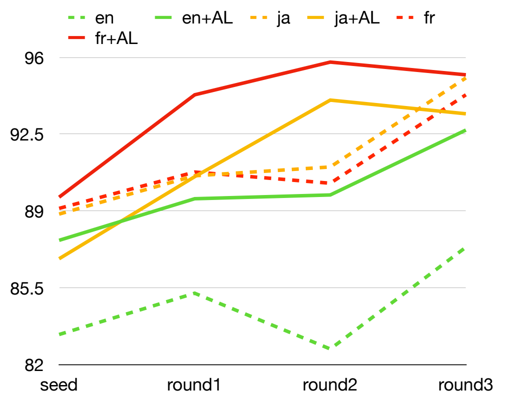
Task	Budget Type	Num Tokens / Instances			AL Details			Num en train
		Train	**Val.**	**Test**	**Seed**	**Val.**	**Budget**	
NER	Token	875k	193k	219k	10k	10k	10k	200k
Classification	Instance	19k	5k	24k	300	300	300	6k
Dependency Parsing	Token	1.88M	196k	189k	17.5k	17.5k	17.5k	350k

Table 4: Aggregate statistics of datasets per task.

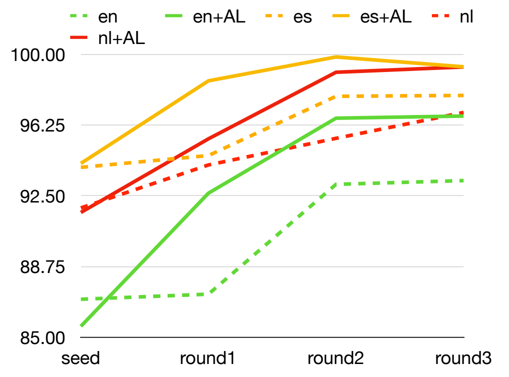
E Active Learning for the MonoA Setup

An interesting observation from Table 3 is that AL in the source language helps improve performance across all languages, irrespective of whether the inference is being run for the source language in question or zero-shot on a different target language without any training. We observe this to be the case consistently for both the NER and the classification tasks (refer Figure 4), regardless of the source language. We hypothesize that this is because the model selects semantically difficult or ambiguous examples that generalize across languages by virtue of mBERT's shared embedding representation, in contrast with random selection where easy examples the model can already tackle might be selected. We observe this even in the case of etymologically distant languages, such as when the model is trained in English and zero-shot inference is done in Japanese (or vice versa). Thus, the AL selection does not overfit on the specific language in question, instead choosing difficult but generalizable examples.

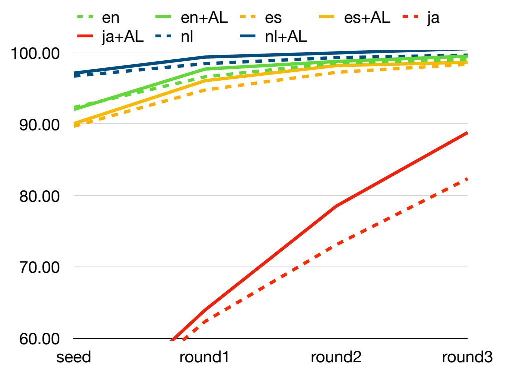
We observe mixed results for the MonoA setup for dependency parsing: AL improves substantially over Random when the target language and the source language are the same; however, when they differ, the results are mixed. We hypothesize that this discrepancy is a consequence of dependency parsing being a syntactic problem, making it more language specific, in turn making zero-shot an inherently harder problem. This is in contrast with both classification and NER, which are more semantic tasks. Consequently, hard examples for the latter tasks might be more generalizable across languages, resulting in their improved AL performance, when compared with the dependency parsing task.



(a) Relative difference of MonoA ± AL for Classification



(b) Relative difference of MonoA ± AL for NER



(c) Relative difference of MonoA ± AL for Dependency Parsing

Figure 4: Performance of mBERT trained on source (de), as a relative percentage of the performance when all source data was used, in a zero-shot classification setting (es and nl).

F Acquisition Ablation Details and Curriculum

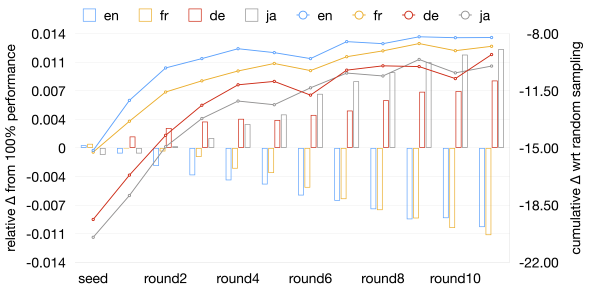


Figure 5: Acquisition curriculum for classification

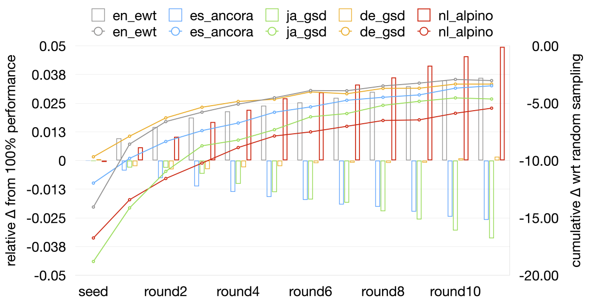


Figure 6: Acquisition Curriculum for dependency parsing. Note that in order to ablate out the effect of different datasets, we only choose the largest dataset for each language.

In this section, we describe the analysis of investigating the acquisitions of SMA+AL in more detail. Let $\alpha_1 \cdots \alpha_n$ be the language specific amount of data present in the entire dataset (i.e $\alpha_i = 0.3$ implies that 30% of the entire dataset (training + unlabeled) is of language i), and let $\beta_{1,1} \cdots \beta_{m,n}$ represent the amount of data acquired for every language at every round (i.e $\beta_{i,j}$ indicates the amount of data acquired by language j at round i). Then, for a task t, for each round i and language j, we plot $\frac{(\sum_{k=1}^{i} \beta_{k,j}) - \alpha_j b_t i}{\alpha_j b_t i}$.

Figures 5 and 6 show the acquisition curriculum. We observe a similar for both the tasks as that for dependency parsing.

G Detailed Results

This section the additional plots as well as the detailed tables and results for all the experiments presented in the paper.

G.1 Per Acquisition Round Performance for Dependency Parsing

Figures 7 and 8 show the UAS and LAS for each round of acquisition for dependency parsing, aggregated across all languages.

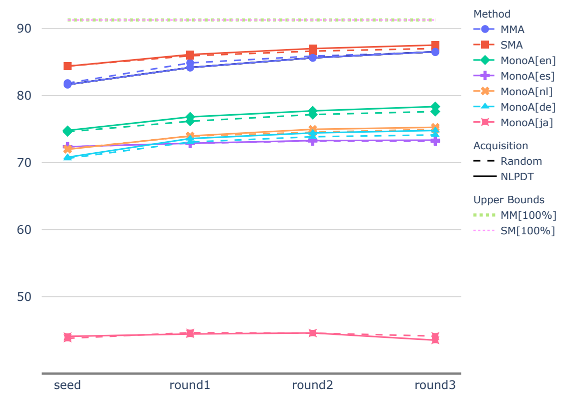


Figure 7: Dependency Parsing: UAS for each round, averaged across all languages

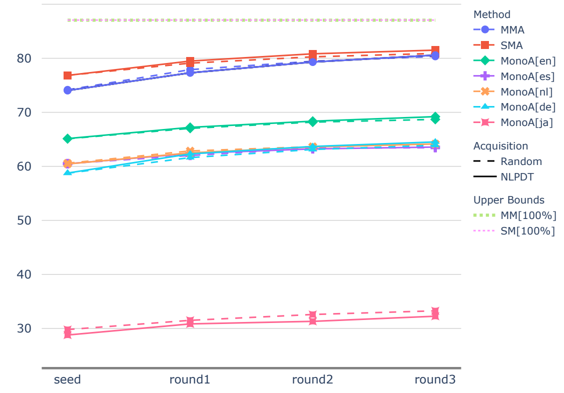


Figure 8: Dependency Parsing: LAS for each round, averaged across all languages

G.2 Per Acquisition Round Performance for NER

Figure 9 shows the F-Score for each round of acquisition for NER, aggregated across all languages.

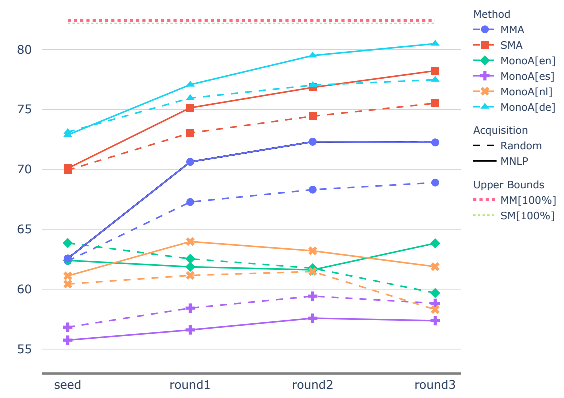


Figure 9: NER: F-Score for each round, averaged across all languages

G.3 Experiments for NER

Tables 5, 6, 7 and 8 show the performance of the different AL settings on English, Spanish, Dutch and German respectively. Each table shows the F-score across 4 acquisition rounds, both with and without MNLP (§3.2).

Acquisition Function		**Without MNLP**				**With MNLP**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	86.0 ± 0.6	87.6 ± 0.2	87.8 ± 0.2	88.4 ± 0.4	85.5 ± 0.4	88.4 ± 0.5	89.2 ± 0.2	89.7 ± 0.5
	de	61.3 ± 1.1	61.5 ± 1.5	65.6 ± 2.2	65.7 ± 1.8	60.3 ± 1.6	65.3 ± 3.2	68.1 ± 1.3	68.2 ± 2.3
	es	55.6 ± 1.1	57.2 ± 1.5	56.7 ± 1.5	58.8 ± 1.7	53.7 ± 1.1	56.8 ± 2.7	57.8 ± 3.0	59.5 ± 2.6
	nl	64.8 ± 3.9	64.7 ± 1.1	67.5 ± 0.6	65.7 ± 1.6	67.8 ± 1.6	68.2 ± 2.0	66.4 ± 2.2	66.0 ± 2.4
MMA		81.9 ± 1.4	84.6 ± 0.5	85.3 ± 1.3	86.5 ± 0.7	82.5 ± 0.4	86.1 ± 0.6	87.4 ± 0.6	88.2 ± 0.5
SMA		82.5 ± 0.6	84.8 ± 0.9	85.8 ± 0.4	86.2 ± 0.3	81.9 ± 0.4	86.6 ± 0.6	87.7 ± 0.5	88.4 ± 0.2
MM Full		91.2 ± 0.2							
SM Full		91.2 ± 0.2							

Table 5: Performance (F1-Score) on en for NER

Acquisition Function		**Without MNLP**				**With MNLP**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	63.0 ± 1.3	64.7 ± 1.2	64.6 ± 1.1	65.4 ± 1.2	63.1 ± 1.7	66.0 ± 1.2	65.9 ± 1.3	67.3 ± 1.0
	de	63.2 ± 0.5	63.6 ± 0.7	65.7 ± 0.2	65.7 ± 0.6	63.3 ± 1.2	66.3 ± 0.8	67.1 ± 0.7	66.8 ± 0.5
	es	76.5 ± 0.6	79.6 ± 0.7	80.2 ± 0.6	81.5 ± 0.5	75.9 ± 0.5	81.0 ± 0.6	82.2 ± 0.5	83.5 ± 0.3
	nl	62.2 ± 1.0	64.3 ± 1.2	67.2 ± 1.1	66.2 ± 1.4	63.0 ± 1.6	67.6 ± 1.0	68.8 ± 1.4	69.8 ± 1.5
MMA		67.8 ± 1.1	71.4 ± 1.7	74.9 ± 2.4	76.1 ± 2.2	68.1 ± 1.3	73.0 ± 1.9	77.4 ± 0.5	78.4 ± 1.2
SMA		73.1 ± 1.0	76.5 ± 0.7	77.9 ± 0.7	79.6 ± 0.3	72.2 ± 0.9	77.7 ± 0.7	79.5 ± 0.3	80.7 ± 0.5
MM Full		86.2 ± 0.7							
SM Full		86.2 ± 0.5							

Table 6: Performance (F1-Score) on es for NER

Acquisition Function		**Without MNLP**				**With MNLP**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	63.9 ± 1.5	62.5 ± 1.0	61.8 ± 1.3	59.7 ± 3.3	62.4 ± 1.4	61.9 ± 1.3	61.6 ± 2.1	63.8 ± 3.3
	de	73.1 ± 0.5	76.9 ± 0.5	77.0 ± 1.2	77.5 ± 0.6	72.9 ± 0.6	77.1 ± 1.1	79.5 ± 0.5	80.5 ± 0.3
	es	56.8 ± 1.1	58.4 ± 1.7	59.4 ± 2.1	58.8 ± 1.6	55.8 ± 2.1	56.6 ± 2.1	57.6 ± 1.3	57.4 ± 1.9
	nl	60.4 ± 1.7	61.2 ± 1.7	61.5 ± 1.1	58.3 ± 3.6	61.1 ± 1.7	64.0 ± 1.3	63.2 ± 2.6	61.9 ± 1.7
MMA		62.4 ± 3.6	67.3 ± 1.2	68.3 ± 1.6	68.9 ± 1.9	62.6 ± 0.6	70.6 ± 1.6	72.3 ± 0.9	72.2 ± 0.8
SMA		69.9 ± 0.8	73.0 ± 0.7	74.4 ± 0.6	75.5 ± 0.6	70.1 ± 0.6	75.1 ± 0.5	76.8 ± 0.2	78.2 ± 0.5
MM Full		82.4 ± 0.5							
SM Full		82.2 ± 0.3							

Table 7: Performance (F1-Score) on de for NER

Acquisition Function		**Without MNLP**				**With MNLP**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	70.9 ± 0.6	72.1 ± 0.9	71.1 ± 1.2	71.2 ± 1.1	71.3 ± 1.6	71.4 ± 1.1	73.1 ± 1.1	73.4 ± 1.5
	de	69.0 ± 1.8	70.7 ± 0.9	71.8 ± 0.5	72.8 ± 0.6	68.8 ± 2.4	71.8 ± 1.0	74.4 ± 0.8	74.6 ± 0.6
	es	61.6 ± 1.2	62.0 ± 1.5	62.2 ± 1.6	63.5 ± 2.7	62.9 ± 1.3	63.0 ± 1.7	62.8 ± 0.8	62.9 ± 1.6
	nl	82.2 ± 0.5	84.5 ± 0.5	85.2 ± 0.4	85.4 ± 0.6	81.6 ± 1.0	86.8 ± 0.4	88.1 ± 0.4	89.0 ± 0.7
MMA		73.1 ± 1.2	76.4 ± 1.4	77.7 ± 0.5	79.3 ± 1.7	72.0 ± 1.8	78.8 ± 1.4	83.0 ± 0.7	84.7 ± 1.1
SMA		79.8 ± 0.7	82.3 ± 0.3	82.3 ± 0.6	82.8 ± 1.0	79.2 ± 0.8	83.1 ± 0.7	85.1 ± 0.3	86.1 ± 0.2
MM Full		90.0 ± 0.5							
SM Full		89.2 ± 1.2							

Table 8: Performance (F1-Score) on nl for NER

G.4 Experiments for Classification

Tables 9, 10, 11 and 12 show the performance of the different AL settings on English, French, Japanese and German respectively. Each table shows the accuracy across 4 acquisition rounds, both with and without LC (§3.2).

Acquisition Function		**Without LC**				**With LC**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	74.8 ± 1.2	79.9 ± 0.6	80.7 ± 1.3	81.5 ± 0.2	76.3 ± 1.3	79.9 ± 1.6	81.6 ± 1.4	82.9 ± 1.8
	fr	65.1 ± 5.5	68.9 ± 5.1	72.2 ± 2.9	71.4 ± 4.3	61.3 ± 4.1	64.1 ± 5.6	71.5 ± 5.4	73.3 ± 4.2
	ja	65.7 ± 4.8	66.5 ± 3.8	68.1 ± 3.4	67.1 ± 4.6	63.9 ± 5.9	70.3 ± 3.3	71.7 ± 2.5	70.1 ± 4.6
	de	58.1 ± 1.6	59.4 ± 3.0	57.7 ± 2.6	60.9 ± 4.2	61.1 ± 3.8	62.4 ± 5.1	62.6 ± 6.8	64.6 ± 6.0
MMA		67.1 ± 1.5	71.0 ± 3.7	74.0 ± 4.1	75.1 ± 2.2	67.4 ± 2.9	72.4 ± 3.4	76.0 ± 3.1	76.6 ± 3.6
SMA		73.5 ± 2.1	76.5 ± 0.5	76.7 ± 0.6	77.6 ± 0.8	71.5 ± 3.3	76.9 ± 1.7	78.6 ± 1.4	79.1 ± 0.8
MM Full		86.6 ± 0.3							
SM Full		87.5 ± 0.5							

Table 9: Performance (Accuracy) on en for Sentiment Classification

Acquisition Function		**Without LC**				**With LC**			
Model	Data %	**5%**	**10%**	**15%**	**20%**	**5%**	**10%**	**15%**	**20%**
MonoA	en	73.1 ± 1.7	75.0 ± 1.4	74.9 ± 2.6	75.7 ± 1.5	74.1 ± 0.6	73.6 ± 2.7	75.5 ± 2.9	76.0 ± 2.2
	fr	75.5 ± 2.6	80.6 ± 0.8	81.7 ± 1.0	83.0 ± 0.8	74.5 ± 1.3	81.4 ± 0.9	82.6 ± 0.5	84.2 ± 0.6
	ja	67.5 ± 4.0	68.7 ± 3.2	68.8 ± 2.4	68.6 ± 2.9	64.9 ± 4.8	71.2 ± 2.8	70.7 ± 1.7	69.8 ± 4.1
	de	64.6 ± 1.2	65.8 ± 3.9	65.4 ± 3.5	68.4 ± 2.3	65.0 ± 2.4	68.4 ± 2.4	69.4 ± 3.2	69.0 ± 4.9
MMA		61.7 ± 3.2	69.9 ± 3.4	73.9 ± 3.0	75.7 ± 2.5	66.0 ± 1.9	71.7 ± 2.1	76.3 ± 0.7	76.6 ± 1.6
SMA		74.6 ± 1.7	77.4 ± 1.2	77.3 ± 0.7	79.2 ± 0.6	72.9 ± 3.1	77.0 ± 1.7	78.4 ± 0.7	79.2 ± 0.8
MM Full		87.8 ± 0.6							
SM Full		89.4 ± 0.4							

Table 10: Performance (Accuracy) on fr for Sentiment Classification

Acquisition Function		Without LC				With LC			
Model \ Data %		5%	10%	15%	20%	5%	10%	15%	20%
MonoA	en	65.6 ± 3.2	65.7 ± 3.5	64.6 ± 3.7	64.8 ± 2.04	68.7 ± 3.2	66.0 ± 3.2	67.2 ± 2.7	66.1 ± 3.10
	fr	67.1 ± 2.5	69.9 ± 2.1	70.2 ± 3.2	70.9 ± 1.46	65.7 ± 1.7	71.3 ± 1.3	68.6 ± 3.6	71.4 ± 1.86
	ja	73.0 ± 3.2	75.4 ± 2.4	77.1 ± 1.2	78.8 ± 1.11	71.9 ± 2.4	77.4 ± 0.8	78.6 ± 1.2	79.5 ± 0.68
	de	64.2 ± 1.8	65.5 ± 4.3	65.8 ± 4.6	68.7 ± 1.82	62.7 ± 2.1	65.4 ± 1.6	68.0 ± 3.5	67.5 ± 4.70
MMA		63.6 ± 1.7	68.0 ± 1.8	70.0 ± 1.5	71.8 ± 0.32	63.6 ± 3.0	67.7 ± 1.6	70.3 ± 1.0	70.6 ± 3.14
SMA		65.8 ± 4.3	69.8 ± 2.0	71.1 ± 2.0	71.6 ± 1.01	65.2 ± 2.0	70.1 ± 2.0	72.2 ± 1.1	72.6 ± 1.71
MM Full		83.7 ± 0.3							
SM Full		84.0 ± 0.2							

Table 11: Performance (Accuracy) on ja for Sentiment Classification

Acquisition Function		Without LC				With LC			
Model \ Data %		5%	10%	15%	20%	5%	10%	15%	20%
MonoA	en	66.7 ± 2.1	69.4 ± 2.0	68.8 ± 1.9	69.1 ± 2.2	68.5 ± 1.4	67.9 ± 3.3	70.0 ± 2.1	72.0 ± 2.5
	fr	67.0 ± 1.6	71.7 ± 0.8	72.1 ± 1.5	72.6 ± 1.0	67.0 ± 1.6	72.1 ± 0.5	71.3 ± 2.2	73.7 ± 1.1
	ja	63.2 ± 3.1	65.8 ± 2.5	65.9 ± 2.5	65.4 ± 2.4	62.8 ± 3.4	67.1 ± 2.2	67.3 ± 0.8	67.1 ± 3.5
	de	67.5 ± 1.0	72.7 ± 0.8	76.3 ± 0.9	77.8 ± 1.3	67.9 ± 3.8	75.6 ± 1.9	78.2 ± 1.2	79.5 ± 0.7
MMA		55.0 ± 2.1	60.4 ± 1.9	61.9 ± 1.8	64.7 ± 1.4	59.2 ± 1.5	62.8 ± 2.2	63.4 ± 2.7	69.0 ± 2.9
SMA		68.9 ± 1.7	72.0 ± 0.8	74.5 ± 1.0	74.0 ± 1.0	66.3 ± 2.7	73.5 ± 1.0	74.9 ± 1.4	75.2 ± 0.9
MM Full		85.7 ± 0.3							
SM Full		87.0 ± 0.3							

Table 12: Performance (Accuracy) on de for Sentiment Classification

G.5 Experiments for Dependency Parsing

Table 13 compares the performance (LAS and UAS) of the single model trained on all data to the performance of one model trained per language. Table 14 gives the detailed breakdown of each AL setup for each of the dependency parsing datasets, aggregated across all the acquisition rounds.

Model	Metric	en-ewt	en-gum	en-lines	en-partut	es -ancora	es-gsd	de-gsd	nl -alpino	nl- lassysmall	ja- gsd	ja- modern	Avg
MM Full	UAS	92.6	91.3	90.6	93.3	94.1	92.4	89.2	94.4	95.1	95.1	75.9	91.3
	LAS	90.2	88.1	86.2	90.0	91.8	88.9	84.6	92.4	92.4	93.9	58.9	87.1
SM Full	UAS	92.5	91.3	90.8	92.8	94.2	92.6	89.7	94.6	95.0	95.1	75.3	91.3
	LAS	90.1	88.0	86.3	89.8	91.8	89.0	85.2	92.9	92.0	93.8	59.6	87.1

Table 13: Performance on 100% data for Dependency Parsing

Dataset	AL	MonoA										MMA		SMA	
		en		es		de		nl		ja					
		UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
en-ewt	Without NLPDT	89.8†	86.1†	76.1	64.4	77.2	65.9	77.5	66.6	38.7	25.0	85.6	80.4	87.2	82.1
	With NLPDT	90.3†	86.9†	75.5	64.2	77.7	66.2	77.1	65.7	38.1	23.4	85.4	80.5	87.8	83.2
en-gum	Without NLPDT	89.5†	85.3†	77.1	66.0	78.1	68.0	78.0	67.3	37.3	23.8	85.9	80.9	88.0	82.8
	With NLPDT	89.9†	85.9†	76.5	65.9	78.7	68.5	77.9	66.9	37.1	22.5	85.2	80.2	88.0	83.1
en-lines	Without NLPDT	88.9†	84.3†	81.1	69.7	81.5	71.2	82.6	72.6	38.8	24.4	86.0	80.7	87.6	82.4
	With NLPDT	89.1†	84.5†	80.6	69.4	81.8	71.3	82.3	72.1	38.4	22.7	85.4	80.1	88.2	83.0
en-partut	Without NLPDT	91.4†	86.6†	83.4	73.5	82.2	73.0	80.5	72.3	37.6	24.3	87.5	81.5	88.7	82.8
	With NLPDT	90.9†	86.0†	83.3	73.3	82.7	73.3	80.5	72.2	37.5	23.4	86.5	80.6	89.0	83.0
es-ancora	Without NLPDT	82.8	70.8	89.2†	84.6†	81.9	70.1	82.8	69.9	26.7	17.0	85.4	78.6	88.0	81.7
	With NLPDT	82.9	70.1	89.4†	84.7†	82.1	70.6	82.9	69.9	26.4	16.6	84.9	77.9	87.9	81.2
es-gsd	Without NLPDT	83.9	73.4	88.0†	81.5†	83.7	72.1	81.7	69.2	27.4	17.3	84.8	76.6	88.0	81.7
	With NLPDT	83.9	72.9	88.3†	82.0†	84.2	72.7	81.7	68.8	27.1	16.8	84.7	76.9	87.8	81.7
de-gsd	Without NLPDT	84.0	74.4	82.2	71.1	87.2†	81.8†	82.8	71.9	43.9	27.8	84.4	77.8	85.8	79.2
	With NLPDT	84.3	74.6	82.2	71.3	87.6†	82.2†	82.9	71.9	43.4	25.5	84.4	78.0	86.0	79.5
nl-alpino	Without NLPDT	83.0	74.5	82.3	70.1	83.4	74.0	91.8†	88.6†	37.0	21.3	86.1	81.3	87.4	81.4
	With NLPDT	83.2	74.6	82.6	70.3	83.9	74.7	92.1†	89.0†	37.1	21.0	85.8	80.8	87.9	82.0
nl-lassysmall	Without NLPDT	82.1	73.0	82.5	70.8	81.6	71.8	92.6†	88.3†	33.0	20.1	86.9	81.2	88.0	80.9
	With NLPDT	82.2	73.0	82.7	70.8	81.9	72.2	92.8†	88.6†	33.0	18.6	86.4	0.7	88.0	81.1
ja-gsd	Without NLPDT	33.8	17.3	31.6	19.1	33.9	17.1	33.5	13.5	93.1†	91.2†	89.1	85.3	87.4	83.0
	With NLPDT	36.9	18.8	32.8	19.7	34.5	17.6	35.0	14.0	93.7†	91.9†	89.7	86.1	88.2	84.2
ja-modern	Without NLPDT	31.1	14.3	28.5	15.0	31.4	15.2	28.8	10.6	73.7†	57.4†	70.6	53.6	69.8	54.2
	With NLPDT	32.5	15.2	28.8	15.0	32.2	16.4	29.2	10.3	74.0†	57.5†	71.0	54.1	70.0	54.3
Avg.	Without NLPDT	76.4	67.2	72.9	62.3	72.9	61.8	73.9	62.8	44.3	31.8	84.8	78.0	86.0	79.3
	With NLPDT	76.9	67.5	73.0	62.4	73.4	62.3	74.0	62.7	44.2	30.8	84.5	77.8	86.3	79.7

Table 14: Performance on different datasets for dependency parsing. † upper-bounds performance for a particular language (since it assigns the entire budget to that language).

G.6 Language Specific Acquisition Plots

Analogous to Figure 1 in the main paper, each figure in this section presents the performance of the different methods for a specific language and a specific task, at each round of acquisition. The trends observed are fairly consistent: SMA and MMA both do consistently well, with SMA outperforming MMA. MonoA for the specific language does well, but with all other languages performs worse. AL consistently improves performance.

G.6.1 NER

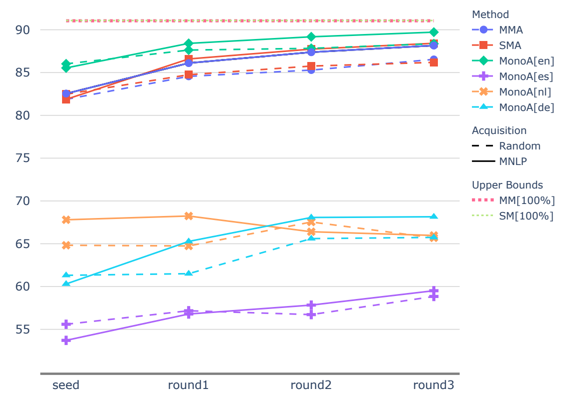


Figure 10: Performance at NER for English (en)

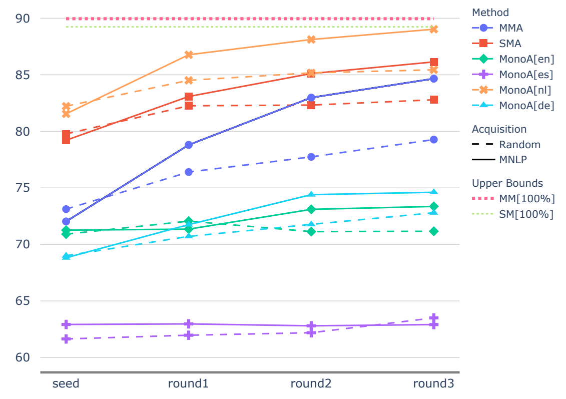


Figure 11: Performance at NER for Dutch (nl)

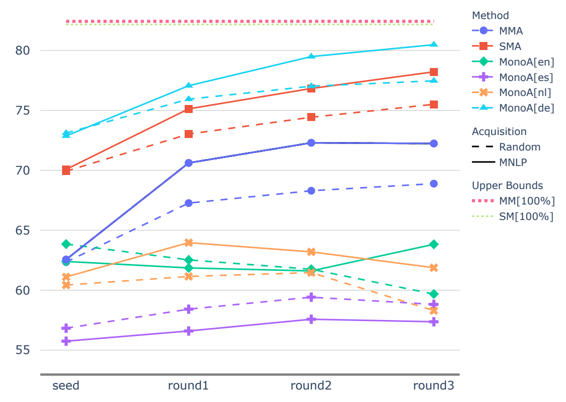


Figure 12: Performance at NER for German (de)

G.6.2 Classification

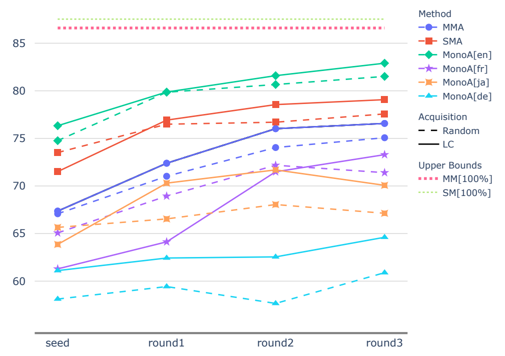


Figure 13: Performance at Classification for English (en)

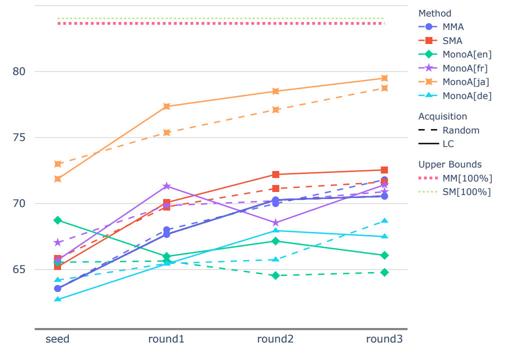


Figure 14: Performance at Classification for Japanese (ja)

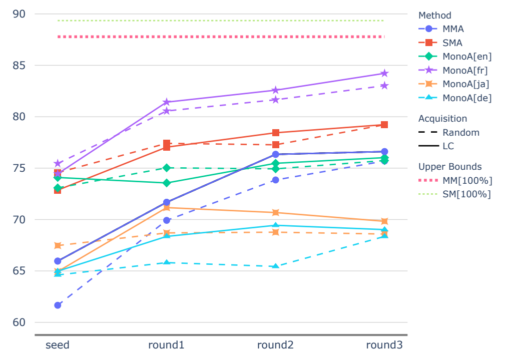


Figure 15: Performance at Classification for French (fr)

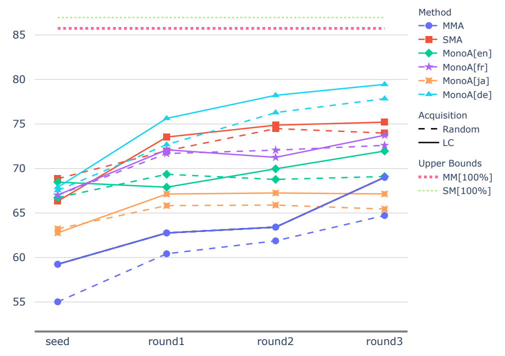


Figure 16: Performance at Classification for German (de)

G.6.3 Dependency Parsing: LAS

For dependency parsing, the MonoA performance of Japanese (MonoA[ja]) is poor on all other languages (Fig. 17, 18, 20, 21, 22, 23, 25, 26), while the performance of all other languages is poor on Japanese (Fig. 19, 24). Consequently, the graphs below have a kink in order to capture this difference in the range of performance of the languages.

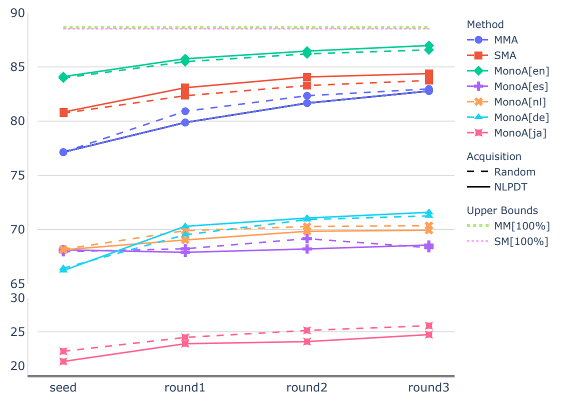


Figure 17: LAS for English (en). Note the kink in the y-axis and the different scales of the two halves.

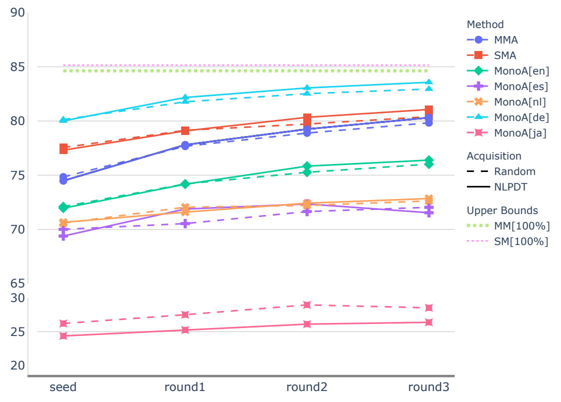


Figure 18: LAS for German (de). Note the kink in the y-axis and the different scales of the two halves.

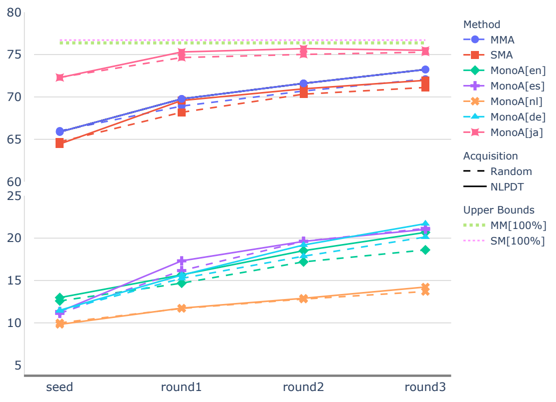


Figure 19: LAS for Japanese (ja). Note the kink in the y-axis and the different scales of the two halves.

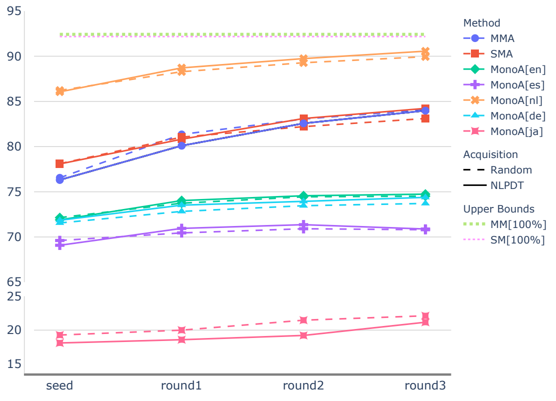


Figure 20: LAS for Dutch (nl). Note the kink in the y-axis and the different scales of the two halves.

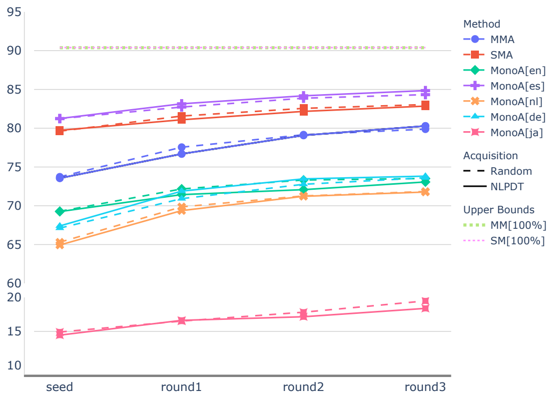


Figure 21: LAS for Spanish (es). Note the kink in the y-axis and the different scales of the two halves.

G.6.4 Dependency Parsing: UAS

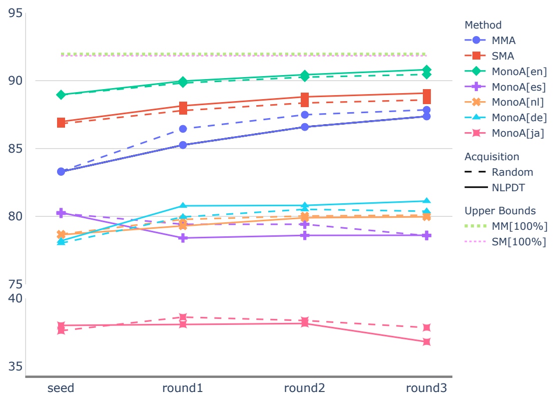


Figure 22: UAS for English (en). Note the kink in the y-axis and the different scales of the two halves.

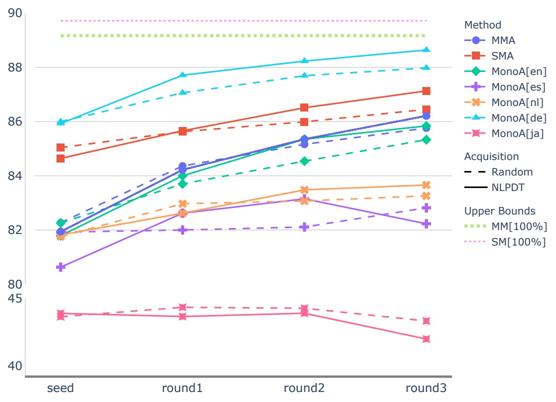


Figure 23: UAS for German (de). Note the kink in the y-axis and the different scales of the two halves.

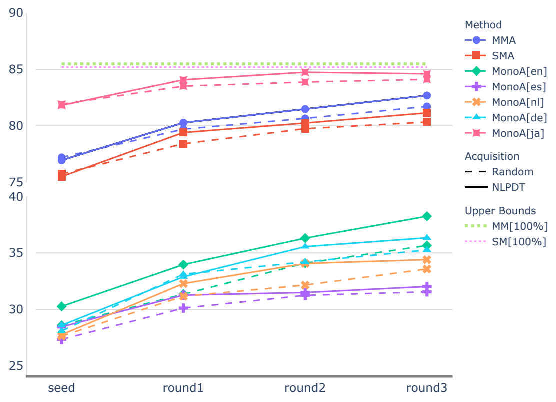


Figure 24: UAS for Japanese (ja). Note the kink in the y-axis and the different scales of the two halves.

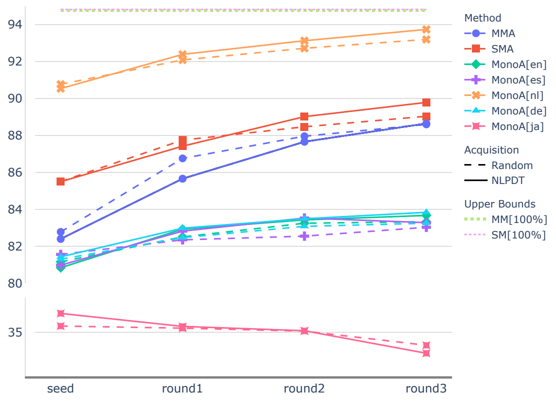


Figure 25: UAS for Dutch (nl). Note the kink in the y-axis and the different scales of the two halves.

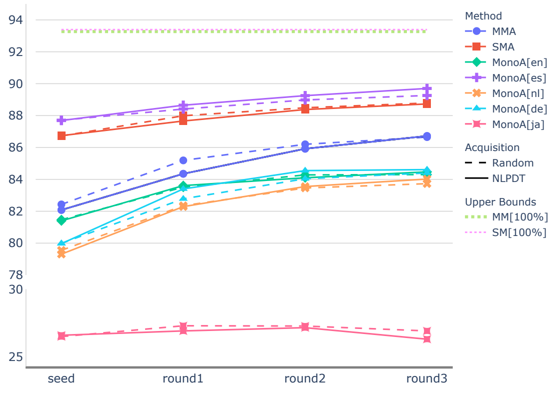


Figure 26: UAS for Spanish (es). Note the kink in the y-axis and the different scales of the two halves.