Domain-Weighted Batch Sampling for Neural Dependency Parsing

Jacob Striebel, Daniel Dakota, Sandra Kübler
Indiana University
{jstrieb,ddakota,skuebler}@indiana.edu

Abstract

In neural dependency parsing, as well as in the broader field of NLP, domain adaptation remains a challenging problem. When adapting a parser to a target domain, there is a fundamental tension between the need to make use of out-of-domain data and the need to ensure that syntactic characteristic of the target domain are learned. In this work we explore a way to balance these two competing concerns, namely using *domain-weighted batch sampling*, which allows us to use all available training data, while controlling the probability of sampling in- and out-of-domain data when constructing training batches. We conduct experiments using ten natural language domains and find that domain-weighted batch sampling yields substantial performance improvements in all ten domains compared to a baseline of conventional randomized batch sampling.

Keywords: neural dependency parsing, domain adaptation, batch optimization

1. Introduction

Dependency parsing, like many other machine learning problems, is sensitive to domain shifts between training and test data sets (Gildea, 2001; Petrov and Klein, 2007). To combat the negative effects of domain shifts when training a parser, several domain adaptation techniques have been studied (e.g., Rosa and Žabokrtský, 2015), although their effectiveness is often limited (e.g., Dredze et al., 2007).

A major factor that determines the success of domain adaptation methods is the amount of training data that is available in the adaptation-target domain (e.g., Daumé III, 2007; Dredze et al., 2007). To overcome the frequent problem of scarcity of target-domain training data, common techniques in parsing focus on selecting optimal source data points to boost performance in the target domain (Plank and van Noord, 2011; McDonald et al., 2011; Mukherjee and Kübler, 2017), with both delexicalized (Rosa and Žabokrtský, 2015) and lexicalized (Falenska and Çetinoğlu, 2017) similarity metrics showing improved data point selection.

Furthermore, to more effectively use all available source- and target-domain data, discrepancies in sizes between data sources have been handled using loss weighting on the different data sources (Dakota et al., 2021), allowing for noise reduction and improved information sharing.

Other approaches for encoding more domain-related information into a parser are to create data- or task-specific embeddings (Stymne et al., 2018; Li et al., 2019, 2020), which yield performance gains across languages and domains. While the further inclusion of language models into parsing architectures noticeably reduces performance gaps across domains, it still cannot fully overcome syntactic differences (Joshi et al., 2018; Fried et al., 2019; Yang et al., 2022). The situation is further complicated by the fact that the source and target domains may be different from those of the language model (Dakota, 2021).

We focus on a setting in which we have access to a small amount of annotated data from the target domain. In order to address the size difference between the data available for the target domain and other domains, we investigate a method that allows the use of all available source and target data during training, thus maximizing the available signal. More specifically, we use *domain-weighted batch sampling* (DWBS) to train a domain-expert neural dependency parser as an alternative to the conventional approach of *randomized batch sampling* (RBS).

Since we use some target domain data for training in our experiments, existing naming conventions are not easily usable. For this reason, we call data from the target domain *in-domain data* and data from all other domains *out-of-domain data* (i.e., any domain that is not the adaptation-target domain); we also use *source data* as a synonym for out-of-domain data. Note that our sampling strategy can also be used when we do not have any in-domain data but can determine the most similar domain among the out-of-domain data.

Our experiments are designed to answer the following two questions:

1. Can we improve parser performance, given a training data imbalance between in-domain and out-of domain data, by replacing the standard batch sampling approach (i.e., RBS) with DWBS, which uses all available training data but favors training sentences drawn from the target evaluation domain?

Proceedings of the Joint Workshop on Multiword Expressions and Universal Dependencies (MWE-UD 2024)
@*LREC-COLING-2024*, pages 198–206. May 25, 2024.

2. Does DWBS yield faster training times than RBS? In other words, does DWBS reduce the number of sample sentences that a parser must observe before dev loss stops decreasing?

2. Domain-Weighted Batch Sampling

2.1. Batch Sampling

When training a neural network, there are several approaches that can be taken to creating batches, and the chosen approach will impact how a network converges, memory requirements, and possible performance among other effects on the model.

The simplest way of creating a batch is to select training samples in the order in which they appear in the training data file, which is called *sequential batch sampling* (SBS). However, this strategy may not be optimal since it repeatedly exposes the network to the same sequence of examples and thus may cause the network to indirectly learn specific batch characteristics that are not representative of the task as a whole (Chollet, 2018), which can result in catastrophic forgetting (French, 1999; Dachapally and Jones, 2018). Consequently, it is more common to create randomized permutations of the training data at the beginning of every epoch, which is called *randomized batch sampling* (RBS).

2.2. Domain-Weighted Batch Sampling

To leverage in-domain and all out-of-domain data, we extend RBS to *domain-weighted batch sampling* (DWBS). This allows for better inclusion of multi-source out-of-domain data, while still permitting the target domain to maintain higher influence on optimization.

To perform DWBS, before training begins the training data set is partitioned into disjoint *in-domain* and *out-of-domain* subsets. For each epoch, random permutations of the in-domain and out-of-domain subsets are separately generated. Each batch is then constructed by drawing sentences (without replacement) from the two permutations until the batch size is reached. We use the hyperparameter μ to define the probability of choosing the next sentence from the in-domain permutation. For example, if μ is equal to 0.45, there is a 45% chance of drawing the next sentence from the target (in-domain) permutation and 55% of drawing from the source (out-of-domain) permutation.

During an epoch, eventually we will attempt to draw from a permutation in which no sentences remain, at which point the current partially constructed batch is discarded and the current epoch is complete. A side-effect of the DWBS procedure is that different epochs may have different durations in terms of number of batches.

Hyperparameter	Value
Optimizer	Adamw
β_1, β_2	0.9, 0.99
Correction bias	False
Learning rate	0.0001
Weight decay	0.01
Gradient normalization	1
LR scheduler	Slanted triangular
Cut fraction	0.2
Decay factor	0.38
Discriminative fine tuning	True
Gradual unfreezing	True
Batch size	32
Patience batches	200
Max steps	153,600
Embeddings	bert-base-cased
Embeddings dim	768

Table 1: Hyperparameters

3. Methodology

3.1. Data

We use Universal Dependency treebanks version 2.12 (Nivre et al., 2020; de Marneffe et al., 2021), more specifically the English Web Treebank (EWT; Bies et al., 2012) and the Georgetown University Multilayer Corpus (GUM; Zeldes, 2017). EWT consists of five domains, and GUM consists of eleven domains.

From the sixteen domains of EWT and GUM, we select only the ten domains that each have a minimum of 1 000 sentences, to limit negative effects during training due to different data sizes across domains. This includes all five of the EWT domains: answers, email, newsgroup, reviews, weblogs; and five from GUM: conversation, fiction, interviews, vlog and whow. We then randomly sub-sample only 1 000 sentences from each domain to create a balanced data set.

All of our experiments use ten-fold cross validation, where, for each fold, each domain is split into 800 train, 100 dev, and 100 test sentences. Consequently, when training each domain-expert parser, there are a total of 8 000 train sentences (800 in-domain and 7 200 out-of-domain), and 100 dev and 100 test sentences (all of these in-domain).

3.2. Parser

We use the deep biaffine attention neural dependency parser (Dozat and Manning, 2017) in the implementation by van der Goot et al. (2021b), which we have modified to allow for DWBS. When training the parser, we use the default hyperparameters provided by van der Goot et al., with the only exception being that we specify early-stopping patience

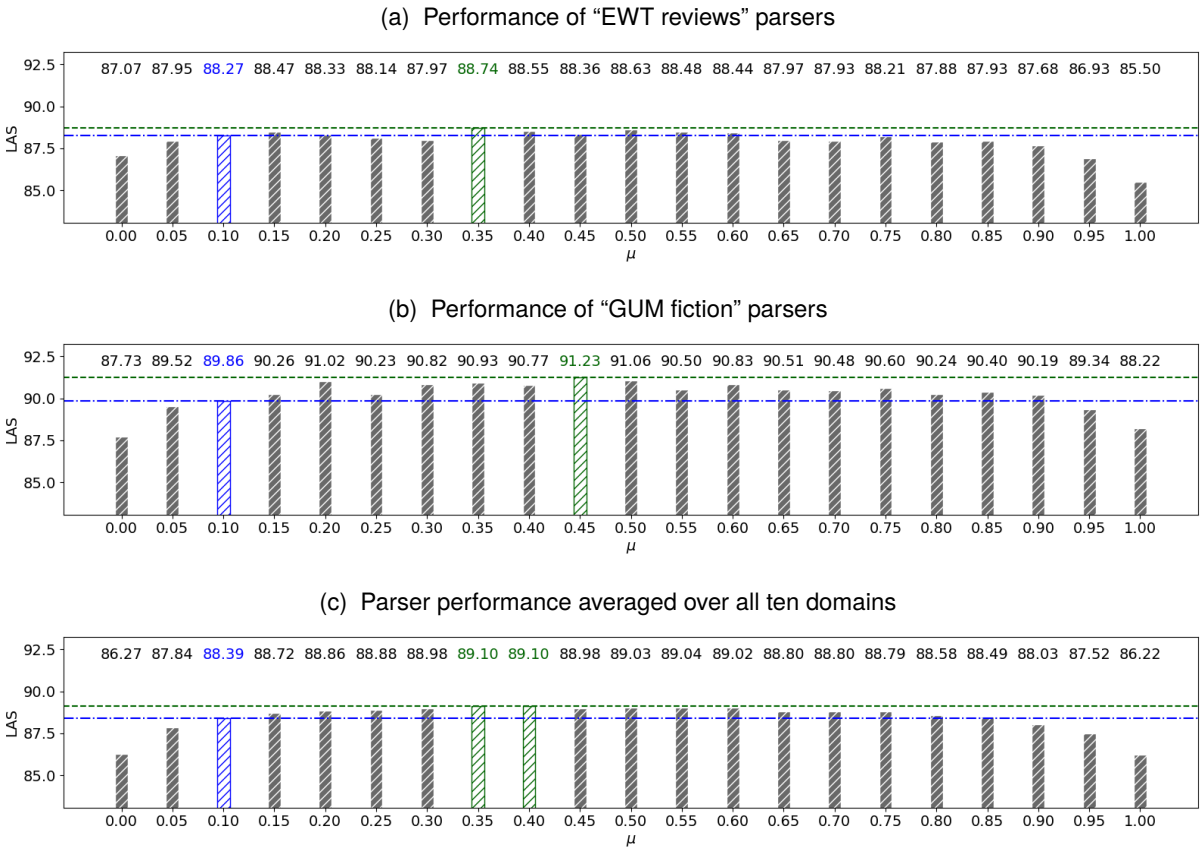


Figure 1: Performance of the DWBS-trained domain-expert parsers on "EWT reviews" (a), "GUM fiction" (b), and averaged over all ten domains (c). X-axis: domain-weight hyperparameter μ; y-axis: parser performance in LAS. Because in our experimental setup we use use ten domains of equal size, whenever μ = 0.10, DWBS is equivalent to conventional RBS; therefore, in each chart we highlight the baseline RBS-trained parser in blue, and we highlight the best performing DWBS-trained parser(s) in green.

in terms of batches rather than epochs, because, when DWBS is enabled, epoch duration varies with μ and it is also subject to random variation (see Section 2.2). Batch size, on the other hand, is a fixed hyperparameter. All hyperparameters are reported in Table 1.

For each domain, and for each of the ten data folds, we use the dev sentences to determine when to stop training, and we then use the test sentences to evaluate. We evaluate using the scorer from the CoNLL 2018 shared task (Zeman et al., 2018).

4. Results

In order to evaluate the effectiveness of DWBS, we perform experiments in which we compare a baseline model trained using conventional RBS against domain-expert parsers trained using DWBS. For each domain, we train domain-expert parsers, with the domain-weight hyperparameter μ ranging from 0.00 to 1.00 (inclusive), with a step size of 0.05. Remember that μ = 0.00 means that each batch will be sampled exclusively from the *out-of-domain* par-

TB	Domain	μ	LAS R	LAS DW
EWT	Answers	0.35	86.78	87.56
	Email	0.35	86.70	**88.00**
	Newsgr.	0.40	88.64	89.44
	Reviews	0.35	88.27	88.74
	Weblog	0.25	89.52	**90.56**
GUM	Convers.	0.35	85.41	**86.64**
	Fiction	0.45	89.86	**91.23**
	Interv.	0.50	88.08	**89.14**
	Vlog	0.60	87.74	88.57
	Whow	0.35	90.46	91.11

Table 2: Performance in LAS per domain, comparing the baseline parser (trained using RBS) to the highest-LAS-producing domain-expert parser (trained using DWBS). LAS R: baseline parser trained using RBS; LAS DW: highest-LAS-producing domain-expert parser trained using DWBS; μ: setting resulting in the highest LAS for the given domain. Improvements of more than 1.00 LAS are bolded.

Treeb.	Domain	μ	RBS NSC	DWBS NSC	ΔNSC
EWT	Answers	0.35	40.40	40.88	0.48
	Email	0.35	39.84	40.00	0.16
	Newsgroup	0.40	45.60	45.44	-0.16
	Reviews	0.35	40.96	41.36	0.40
	Weblog	0.25	47.04	48.00	0.96
GUM	Conversation	0.35	45.52	41.20	-4.32
	Fiction	0.45	40.56	42.96	2.40
	Interview	0.50	45.20	42.00	-3.20
	Vlog	0.60	48.16	42.96	-5.20
	Whow	0.35	40.40	40.24	-0.16

Table 3: Training duration per domain measured in number of thousands of samples until model convergence, comparing the baseline parser to the highest-LAS-producing domain-expert parser. NSC: number of thousands of training samples until model convergence; RBS NSC: NSC for the baseline parser trained using RBS; DWBS NSC: NSC for the highest-LAS-producing domain-expert parser trained using DWBS; μ: setting yielding the best (in terms of LAS) domain-expert parser for the given domain.

tition of the training data set, while μ = 1.00 means that training samples will only be drawn from the *in-domain* partition. Because our training data set is composed of ten domains of equal size, DWBS for μ = 0.10 is equivalent to conventional RBS.

4.1. Effect on Parsing Accuracy

The DWBS-trained parser outperforms the baseline in all ten domains tested, for some settings of μ. We provide full results for two domains, plus the results averaged over all ten domains, in Figure 1; full results for the remaining domains are supplied in Appendix A. Table 2 summarizes the results by giving the LAS for the highest performing DWBS-trained parser, per domain, and giving the setting for μ that produced the parser.

The domain which benefits least from DWBS, in terms of absolute increase in LAS over the baseline, is EWT reviews, for which the best setting of μ = 0.35 yields an improvement of 0.47 LAS (see Figure 1a); the domain benefiting most is GUM fiction, for which the best setting of μ = 0.45 gives an improvement of 1.37 LAS (see Figure 1b). The average improvement across all ten domains, using each domain's best setting of μ, is 0.95 LAS. As shown in Table 2, five domains experience gains of more than 1.00 LAS.

Overall, the best setting of μ ranges between 0.25 (EWT weblog) and 0.60 (GUM vlog). GUM domains tends to prefer higher values of μ. In other words, those domains profit more from training examples from the same domain, which is an indication that each of those domains is different from all others, either in terms of syntactic structure or annotation.

4.2. Effect on Training Duration

Our hypothesis wrt training times is that the more target-domain sentences that are included in training batches, the faster the parser should converge, since the training sentences should be more consistent and also more similar to the dev data. This hypothesis is supported by findings that alternative batch sampling techniques to RBS which are similarly motivated to DWBS yield significantly faster network training times on several tasks (Loshchilov and Hutter, 2016).

We show the average number of training examples until model convergence for the highest-LAS-producing μ per domain in Table 3. In contrast to the results presented in the previous subsection in which all ten domains show an improvement in LAS, the domains are evenly split on training time reduction with five seeing a reduction and five experiencing an increase. The greatest increase is experienced by the GUM fiction domain, which requires 2400 more sentences than the baseline to achieve parser convergence, while the greatest decrease is experienced by the GUM vlog domain, which shows a decrease of 5200 sentences until convergence. The average change in training samples is a decrease of 864 sentences. The high variability of differences in training duration suggests that DWBS does not reliably reduce the number of samples required to achieve parser convergence. This may suggest that our target domain data do not always have high internal consistency, which is in line with findings by Zeldes and Schneider (2023), who observed considerable differences in cross-domain parsing between EWT and GUM.

Interestingly, four out of the five domains showing decreased training times are GUM domains. Since GUM domains also prefer higher values of μ, this could suggest that sampling more target sentences reduces training time.

5. Conclusion

In this work we investigated the effectiveness of domain-weighted batch sampling (DWBS) when training a neural dependency parser. DWBS is a technique for constructing training batches that can be used in cases when the domain that a parser will be evaluated on is known and there is also training data available in the evaluation domain. We conducted experiments using ten English domains and found that DWBS produced higher performing parsers than RBS in all ten domains. This finding suggests that when the preconditions for performing DWBS are met, it should be preferred to RBS when training a neural dependency parser.

The success of DWBS for neural dependency parsing suggests several directions for future work: In the present experiment while training each model, the domain-weight parameter μ was held constant for the full duration of training. An alternative is to begin training with μ equal to the baseline setting, and then gradually increase μ as training progresses. This will simulate *gradually fine-tuning* the parser in the target domain. A second area of future work is to experiment with methods of automatically classifying domains (e.g., in the style of Mukherjee et al., 2017; Mukherjee and Kübler, 2017), which would allow for the discovery of more syntactically useful domain groupings. Finally, we will investigate the effectiveness of domain embeddings (van der Goot and de Lhoneux, 2021; van der Goot et al., 2021a; Li et al., 2019, 2020), an alternative approach to domain adaptation in dependency parsing that can be combined with domain-weighted batch sampling.

6. Acknowledgments

The authors acknowledge the Indiana University Pervasive Technology Institute for providing supercomputing and storage resources that have contributed to the research results reported within this paper.

This research was supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute.

This research is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract #2022-22072200002. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

7. Bibliographical References

Francois Chollet. 2018. *Deep Learning with Python*. Manning, Shelter Island, NY.

Prudhvi Dachapally and Michael Jones. 2018. Catastrophic interference in neural embeddings models. In *Proceedings of the 40th Annual Meeting of the Cognitive Science Society*.

Daniel Dakota. 2021. Genres, parsers, and BERT: The interaction between parsers and BERT models in cross-genre constituency parsing in English and Swedish. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 59–71, Online. Association for Computational Linguistics.

Daniel Dakota, Zeeshan Ali Sayyed, and Sandra Kübler. 2021. Bidirectional domain adaptation using weighted multi-task learning. In *Proceedings of the 17th International Conference on Parsing Technologies and the IWPT 2021 Shared Task on Parsing into Enhanced Universal Dependencies (IWPT 2021)*, pages 93–105, Online. Association for Computational Linguistics.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *International Conference on Learning Representations*.

Mark Dredze, John Blitzer, Partha Pratim Talukdar, Kuzman Ganchev, João Graca, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1051–1055, Prague, Czech Republic.

Agnieszka Falenska and Özlem Çetinoğlu. 2017. Lexicalized vs. delexicalized parsing in low-resource scenarios. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 18–24, Pisa, Italy.

Robert French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3:128–135.

Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. Cross-domain generalization of neural constituency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 323–330, Florence, Italy.

Daniel Gildea. 2001. Corpus Variation and Parser Performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 167–202, Pittsburgh, PA.

Vidur Joshi, Matthew Peters, and Mark Hopkins. 2018. Extending a parser to distant domains using a few dozen partially annotated examples. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1190–1199, Melbourne, Australia.

Ying Li, Zhenghua Li, and Min Zhang. 2020. Semi-supervised domain adaptation for dependency parsing via improved contextualized word representations. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3806–3817, Barcelona, Spain (Online).

Zhenghua Li, Xue Peng, Min Zhang, Rui Wang, and Luo Si. 2019. Semi-supervised domain adaptation for dependency parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2386–2395, Florence, Italy.

Ilya Loshchilov and Frank Hutter. 2016. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72, Edinburgh, Scotland, UK.

Atreyee Mukherjee and Sandra Kübler. 2017. Similarity based genre identification for POS tagging experts & dependency parsing. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 519–526, Varna, Bulgaria. INCOMA Ltd.

Atreyee Mukherjee, Sandra Kübler, and Matthias Scheutz. 2017. Creating POS tagging and dependency parsing experts via topic modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 347–355, Valencia, Spain. Association for Computational Linguistics.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC)*, pages 4034–4043, Marseille, France.

Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 404–411, Rochester, NY.

Barbara Plank and Gertjan van Noord. 2011. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1566–1576, Portland, Oregon, USA.

Rudolf Rosa and Zdeněk Žabokrtský. 2015. KL-cpos3 - a language similarity measure for delexicalized parser transfer. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 243–249, Beijing, China.

Sara Stymne, Miryam de Lhoneux, Aaron Smith, and Joakim Nivre. 2018. Parser training with heterogeneous treebanks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 619–625, Melbourne, Australia. Association for Computational Linguistics.

Rob van der Goot and Miryam de Lhoneux. 2021. Parsing with pretrained language models, multiple datasets, and dataset embeddings. In *Proceedings of the 20th International Workshop on Treebanks and Linguistic Theories (TLT, SyntaxFest 2021)*, pages 96–104, Sofia, Bulgaria. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, and Barbara Plank. 2021a. On the effectiveness of dataset embeddings in mono-lingual,multi-lingual and zero-shot conditions. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 183–194, Kyiv, Ukraine. Association for Computational Linguistics.

Rob van der Goot, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021b. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics:*

System Demonstrations, pages 176–197, Online. Association for Computational Linguistics.

Sen Yang, Leyang Cui, Ruoxi Ning, Di Wu, and Yue Zhang. 2022. Challenges to open-domain constituency parsing. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 112–127, Dublin, Ireland. Association for Computational Linguistics.

Amir Zeldes and Nathan Schneider. 2023. Are UD treebanks getting more consistent? a report card for English UD. In *Proceedings of the Sixth Workshop on Universal Dependencies (UDW, GURT/SyntaxFest 2023)*, pages 58–64, Washington, D.C.

Daniel Zeman, Jan Hajič, 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, Brussels, Belgium.

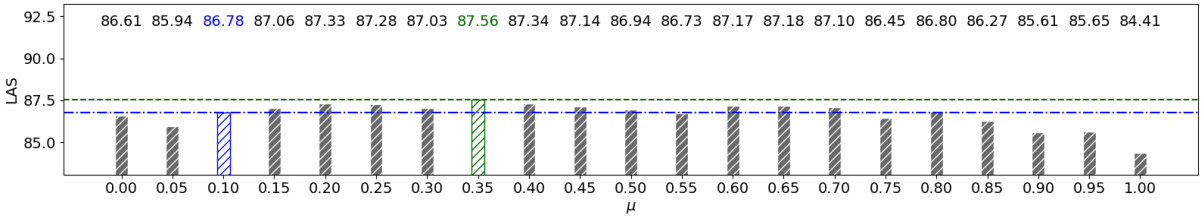
8. Language Resource References

Bies, Ann and Mott, Justin and Warner, Colin and Kulick, Seth. 2012. *English Web Treebank LDC2012T13*. Linguistic Data Consortium.

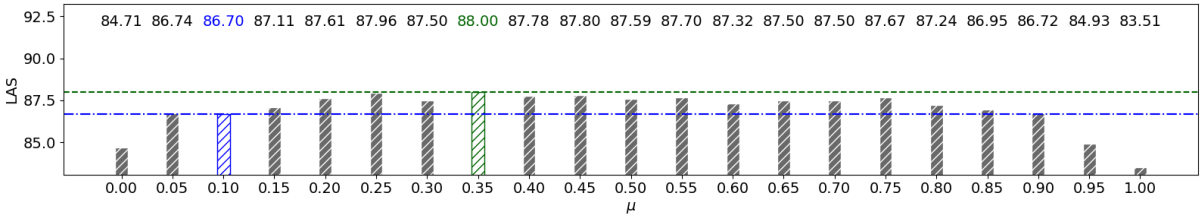
Amir Zeldes. 2017. *Georgetown Multilayer Corpus (GUM)*. Georgetown University Corpus Linguistics Lab.

A. Complete Parsing Results

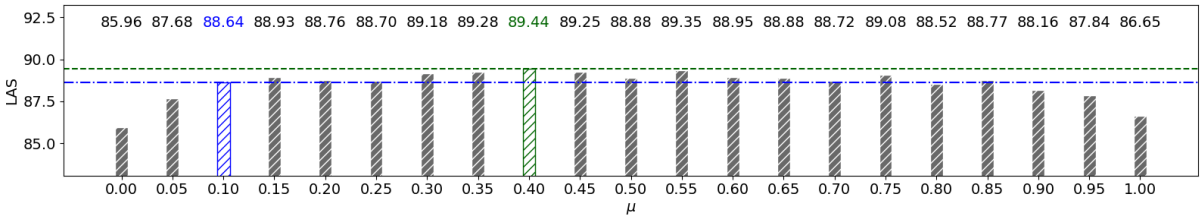
(a) Performance of “EWT answers” parsers



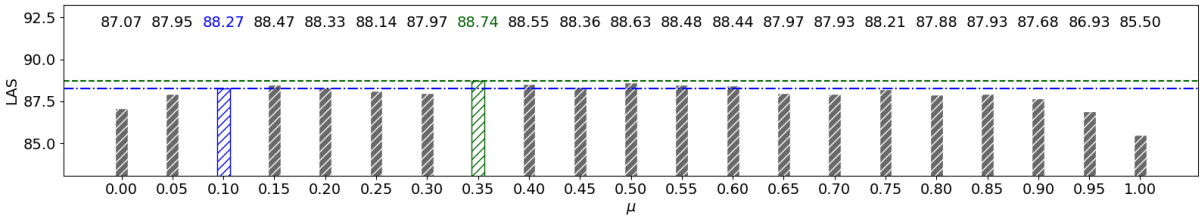
(b) Performance of “EWT email” parsers



(c) Performance of “EWT newsgroup” parsers



(d) Performance of “EWT reviews” parsers



(e) Performance of “EWT weblog” parsers

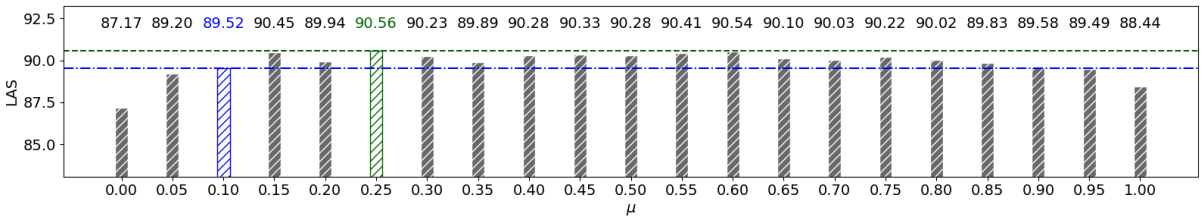


Figure 2: Parser performance in the five **English Web Treebank** domains. X-axis: domain-weight hyperparameter μ; y-axis: parser performance (LAS). Baseline RBS-trained parser in blue, and best performing DWBS-trained parser in green.

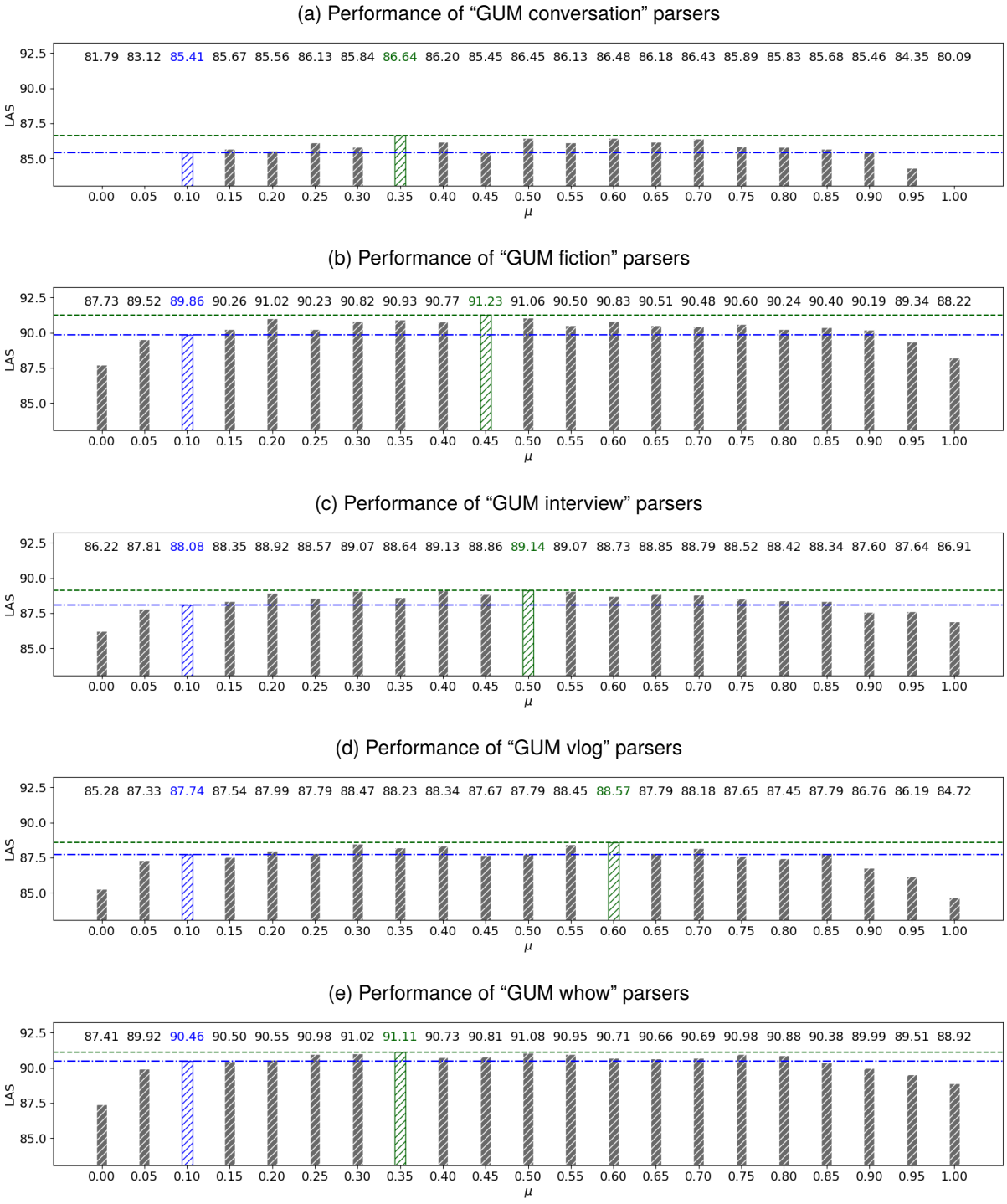


Figure 3: Parser performance in the five **Georgetown University Multilayer Corpus** domains. X-axis: domain-weight hyperparameter μ; y-axis: parser performance (LAS). Baseline RBS-trained parser in blue, and best performing DWBS-trained parser in green.