Pre-training Language Models for Surface Realization

Farhood Farahnak
Concordia University
Montreal, Canada
farhood.farahnak@gmail.com

Leila Kosseim
Concordia University
Montreal, Canada
leila.kosseim@concordia.ca

Abstract

Surface Realization in Natural Language Generation (NLG) is the task of deriving the surface form of a sentence (the actual words) from an underlying representation. Following recent advances in deep learning, several models have been proposed for different NLG sub-tasks including surface realization. Most of these models require a large amount of training data, however, acquiring accurately labeled data is laborious and expensive. In this work, we study how synthetically generated labeled data can be leveraged to improve the performance of a surface realization model. By pre-training a language model on automatically labeled data and then fine-tuning it on manually labeled data, our approach improved the state-of-the-art performance on the standard English datasets from the deep track of the Multilingual Surface Realization (MSR) workshop (Belz et al., 2020) by more than 10% BLEU score.[1]

1 Introduction

The goal of Natural Language Generation (NLG) is to generate text in human languages (e.g. English) for a wide range of applications such as report generation, text summarization, and conversation modeling. NLG involves both *content planning* (selecting the content to communicate) and *surface realization*. Surface realization (SR), the last step of the NLG pipeline, aims to derive the surface form of a sentence (the actual words) from an underlying representation by choosing the proper word forms (inflection, punctuation, and formatting) and determining their correct order (syntactic realization) (Hovy et al., 1996; Reiter and Dale, 2000).

Recent advances in Natural Language Processing (NLP) and Deep Neural Networks (DNN) have led to drastic improvements in many NLP systems, some of which have even achieved human-level performance (Läubli et al., 2018). Similarly to many NLP models, surface realization models have also benefited from these advancements. DNN models usually require a large amount of labeled data for training; however, creating accurate and reliable training data is an expensive and time consuming task. In this work, we show how we can improve the performance of surface realization by pre-training a language model on a large synthetically generated dataset and then fine-tuning it on a smaller manually labeled dataset.

To measure the effectiveness of our approach, we followed the protocol of the Multilingual Surface Realization (MSR) Workshops (Mille et al., 2018, 2019; Belz et al., 2020), and generated the surface form of sentences from their dependency parse trees. To create the synthetic data, we used the automatic dependency parser Stanza (Qi et al., 2020) to parse the unlabeled WikiText corpus (Merity et al., 2017). Using different sizes of manually labeled and synthetic data, we investigated the effects of the proposed pre-training phase. Although the synthetic data may contain noisy annotations compared to manually labeled data and may come from a different distribution (e.g. different textual genre or discourse domain), results show that its sheer size allows the model to learn the general gist of the task in the pre-training phase and leads to an increase in performance in SR achieving state-of-the-art performance on the deep track with the English datasets of the MSR workshops.

2 Background

2.1 Multilingual Surface Realization (MSR)

The Multilingual Surface Realization (MSR) workshops have organized shared tasks aimed at bring-

[1] The code is available at https://github.com/CLaC-Lab/SR_LM

ing together researchers interested in surface oriented Natural Language Generation problems and share resources to that end (Mille et al., 2018, 2019; Belz et al., 2020). The shared task aimed to generate the surface form of sentences given their Universal Dependency (UD) structures. Two tracks were proposed: the shallow and the deep tracks. For the shallow track, word order information and the inflected form of words were removed from the UD structure and the task aimed to determine the correct order of words and inflect them. In the deep track, in addition to word ordering and inflection, functional words (in particular, auxiliaries, functional prepositions and conjunctions) and surface-oriented morphological information were removed from the UD structure and had to be recovered by the models.

2.2 Previous Work

Participants in the Multilingual Surface Realization (MSR) workshops proposed different models to address the surface realization task. Many of these models use dedicated sub-modules for each subtask. For example the ADAPT center (Elder, 2020) proposed a biLSTM sequence-to-sequence model with a copy mechanism to generate the surface form of sentences. They augmented the training set with 4.5M sentences from two sources, WikiText (Merity et al., 2017) and CNN stories (Hermann et al., 2015), and chose sentences that had at least 80% word overlap with the labeled dataset to ensure that they have a similar distribution. The BME-TUW system (Recski et al., 2020) used an Interpreted Regular Tree Grammar to retrieve the correct order of tokens then used a biLSTM sequence-to-sequence model to inflect the words. The IMS system (Yu et al., 2020) tackled the surface realization problem as a Traveling Salesperson Problem, and used a biaffine attention model to calculate the bigram scores for the output sequence. Finally, they used a biLSTM for the inflection module. Similarly to the ADAPT center, IMS also used WikiText and CNN stories to augment their training data with $200K$ synthetic samples, however, by considering the branching factors of the tree, they tried to keep the distribution of the augmented data close to the labeled datasets. The data augmentation that ADAPT and IMS used differ from our proposed solution as they both tried to keep the distribution of the augmented data as similar as possible to the manually labeled data by applying filtering rules. In contrast, our approach does not enforce the distributions to be similar, and lets the domain adaptation to be performed automatically during the fine-tuning phase.

Because of their simplicity and effectiveness, several approaches have used language models for surface realization. The NILC system (Cabezudo and Pardo, 2020) proposed to use GPT-2 (Radford et al., 2019) and linearization using the parentheses approach. We argue that when the number of nodes grows, the model has difficulties in capturing the relations between them. The Concordia system (Farahnak et al., 2020) used BART (Lewis et al., 2020) for surface realization, however, the relation between nodes was represented with the actual words. This approach may cause problems when a word appears more than once in a sentence as the model cannot capture the exact structure of the tree. Our approach is also based on language models, however, it differs from theirs as indices are used to encode the edges in the UD structure instead of the actual tokens (see Section 4).

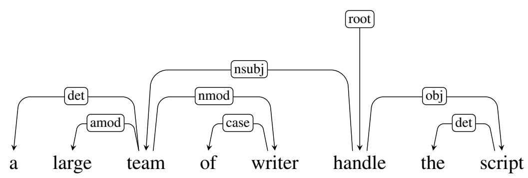

Figure 1: Example of UD dependency parse tree for the sentence *A large team of writers handled the script.*

3 Data

In this section, we first present the MSR manually labeled datasets we used for our experiments and then discuss how we created the synthetic dataset.

3.1 Manually Labeled Datasets

For our experiments, we used the English datasets provided by the MSR workshop (Belz et al., 2020). These datasets are modified versions of the Universal Dependency (UD) datasets (de Marneffe et al., 2014) where the order of the tokens is shuffled and the inflected form of the tokens are removed. Table 1 presents statistics of these datasets. As Table 1 shows, the largest dataset (EWT) contains only $\approx 12K$ training samples which makes it hard to train an DNN model based solely on these samples.

Dataset	train	dev	test
EWT	12,543	2,002	2,077
GUM	2,914	707	778
LinES	2,738	912	914
ParTUT	1,781	156	153

Table 1: Number of samples in the English MSR datasets.

3.2 Synthetically Generated Dataset

In order to generate synthetic data, we used the WikiText dataset (Merity et al., 2017), extracted from Wikipedia articles. The WikiText dataset comes from a different domain compared to the MSR datasets[2] which makes it a suitable candidate to study the domain adaptation between the two text genres. We extracted the first $500K$ sentences after filtering non-English sentences and sentences longer than 150 characters[3] to create our synthetic dataset. We used Stanza (Qi et al., 2020) to parse the sentences and create their UD structure. Using the script provided by the MSR workshop (Belz et al., 2020), we generated the synthetic dataset in the same format as provided by the workshop. Figure 1 shows a visual representation of the dependency tree structure of a sample from the dataset.

4 Model

Following the success of pre-trained language models (LMs) for data-to-text generation tasks (Kale and Rastogi, 2020; Harkous et al., 2020; Farahnak et al., 2020), we used an encoder-decoder LM for surface realization. The input and output of an encoder-decoder LM is in linear form (text-to-text); however, surface realization is a data-to-text task. In order to use LM, the input UD structure had to be linearized. Among the features available in the UD structure, we considered `lemma` (the lemmatized form of tokens), `FEATS` (morphological information), `HEAD` (the parent in the tree structure), and `deprel` (dependency relation to the head) and represented each node in the linear format:

`index : lemma FEATS : head_index <deprel>`

then concatenated all nodes together. Figure 2 shows the linearized representation of the example from Figure 1 used for the shallow track. In this example, the parent of the word `script` is node 5 which is the index for word `handle`. To train the LM, we used the surface form of the sentence as the target. The model learns to generate the surface form given the linearized UD structure, hence, it learns to perform both syntactic and morphological realization simultaneously.

4 : script Sing : 5 <obj> # 3 : writer Plur : 7 <nmod> # 9 : . : 5 <punct> # 6 : large Pos : 7 <amod> # 7 : team Sing : 5 <nsubj> # 5 : handle Ind Plur 3 Past Fin : ROOT <root> # 1 : the Def Art : 4 <det> # 8 : of : 3 <case> # 2 : a Ind Art : 7 <det>

Figure 2: Linearized representation of the UD structure of Figure 1.

5 Experiments and Results

5.1 Experimental Setup

In order to understand the effect of synthetic data on the performance of the ordering model, we conducted several experiments using different sizes of synthetic data to pre-train the model, then fine-tuning it on the manually labeled datasets and measuring the performance on the MSR test sets (see Table 1). For all experiments, we used the pre-trained BART (Lewis et al., 2020) large model. We used the AdamW (Loshchilov and Hutter, 2019) optimization algorithm with a learning rate of $1e$-5 and batch size of 4 to train our models. We pre-trained the models for 5 epochs on the synthetic data and fine-tuned them for 5 more epochs on the manually labeled data. For comparative purposes, we also trained the models without the pre-training phase, and trained them for 15 epochs on the manually labeled data. We choose the model with the highest performance on the development sets.

5.2 Results

Table 2 compares the performance of training the encoder-decoder language model using different sizes of synthetic data for pre-training. Our experiments suggest that the pre-training phase can improve the performance of the model by 3.90% and 5.21% in BLEU score for the shallow and deep tracks respectively on the EWT dataset. However, the improvement on the other three datasets are more significant, ranging from 12.76% to 25.65%, as these datasets have much fewer training samples compared to EWT. The improvement of pre-training on synthetic data is higher for the deep

[2] The EWT dataset contains sentences from five genres of web media: weblogs, newsgroups, emails, reviews, and Yahoo! answers.

[3] This value was chosen because 90% of the samples in EWT are shorter than 150 characters.

		Shallow Track								Deep Track							
		EWT	Δ	GUM	Δ	LinES	Δ	ParTUT	Δ	EWT	Δ	GUM	Δ	LinES	Δ	ParTUT	Δ
	0	80.79	_	71.63	_	69.62	_	67.84	_	64.31	_	48.74	_	40.61	_	49.23	_
# synthetic samples for pre-training	100K	84.38	3.59	86.27	14.64	82.38	12.76	86.98	19.14	68.10	3.79	68.61	19.87	64.28	23.67	69.50	20.27
	200K	84.62	3.83	86.84	15.21	83.00	13.38	86.69	18.85	69.02	4.71	69.21	20.47	65.29	24.65	69.25	20.02
	500K	**84.69**	3.90	**86.76**	15.13	**83.18**	13.56	87.66	19.82	**69.52**	5.21	**70.19**	21.45	**66.26**	25.65	**71.38**	22.15

Table 2: BLEU score of models pre-trained with different sizes of synthetic data. Δ reports the difference of the pre-trained models to training without the pre-training phase (i.e. 0 synthetic data).

		EWT			GUM			LinES			ParTUT		
		BLEU	NIST	DIST	BLEU	NIST	DIST	BLEU	NIST	DIST	BLEU	NIST	DIST
Shallow Track	BME (Recski et al., 2020)	57.25	12.52	65.23	60.77	12.10	62.86	55.98	11.78	61.44	61.37	10.22	58.39
	Concordia (Farahnak et al., 2020)	70.71	12.70	77.94	66.98	11.62	69.87	62.70	11.30	68.62	67.05	9.83	71.59
	IMS (Yu et al., 2020)	85.67	13.74	87.74	**89.70**	**12.98**	**91.97**	**85.30**	**12.97**	**86.48**	**89.37**	**11.05**	**88.73**
	ADAPT (Elder, 2020)	**87.50**	**13.81**	**90.35**	_	_	_	_	_	_	_	_	_
	Our Approach	84.69	13.58	88.82	86.76	12.65	89.12	83.18	12.59	85.72	87.66	10.91	86.80
Deep Track	NILC (Cabezudo and Pardo, 2020)	45.19	9.96	64.83	53.92	9.00	60.42	41.04	9.09	61.18	43.41	8.24	59.74
	Concordia (Farahnak et al., 2020)	58.44	11.61	73.66	53.92	10.51	67.02	47.96	9.93	64.33	50.54	8.57	62.39
	IMS (Yu et al., 2020)	58.66	11.61	79.23	53.92	11.25	76.47	50.45	10.89	73.1	50.11	9.26	72.98
	Our Approach	**69.52**	**12.54**	**82.43**	**70.19**	**11.64**	**80.93**	**66.26**	**11.37**	**78.81**	**71.38**	**9.99**	**77.88**

Table 3: Comparison of our approach (models pre-trained on $500K$ synthetic sentences and fine-tuned on each dataset) with previous models proposed for the deep track of MSR 2020.

track compared to the shallow track as the task is more complex in the sense that the model not only needs to learn the inflection and ordering of words, it also needs to guess the removed functional words.

In comparison with previous participating models of MSR 2020 (Belz et al., 2020) (see Table 3), our approach is not able to outperform the previous work on the shallow track. However, it improves the state-of-the-art performance by a large margin (more than 10% in BLEU score) on in the deep track on all datasets which shows the superiority of our proposed approach.

5.3 Analysis

We analysed the results of the models to better understand the benefits and drawbacks of our approach.

Pre-training seems to facilitate domain adaption, as a single epoch of fine-tuning is enough for the model to adapt to the domain of the manually labeled dataset (see Appendix A.1).

Pre-training can significantly reduce the need for manual data. We fine-tuned the pre-trained models using subsets of the manually labeled data. Results shows that with pre-training, using only 10% of the data achieves better performance than training on all manually labeled data without the pre-training phase (see Appendix A.2).

Finally, through a manual inspection of the generated sentences (see Appendix A.3), we determined that most errors should actually be considered correct alternatives to the ground truth. Better automatic measures should be developed to measure the performance of surface realization to account for linguistic variations.

6 Conclusion and Future Work

In this paper, we showed that pre-training on synthetic data is beneficial for surface realization even when the data comes from a different distribution than the training data. We also showed that the pre-training phase not only improves the performance of the model, but also helps the model to converge faster on the training data. The proposed pre-training phase for LM improved the state-of-the-art performance on the standard English datasets from the deep track of the MSR workshop (Belz et al., 2020) by more than 10% BLEU score.

As of future work, we plan to conduct similar experiments on previously proposed models such as ADAPT (Elder, 2020) and IMS (Yu et al., 2020). We also plan to run cross-language experiments to see whether the knowledge learned from one language can be transferred to another language.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments on an earlier version of this paper. This work was financially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

Anya Belz, Bernd Bohnet, Thiago Castro Ferreira, Yvette Graham, Simon Mille, and Leo Wanner. 2020. Proceedings of the Third Workshop on Multilingual Surface Realisation. Barcelona, Spain (Online). Association for Computational Linguistics.

Marco Antonio Sobrevilla Cabezudo and Thiago Pardo. 2020. NILC at SR'20: Exploring pre-trained models in surface realisation. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 50–56, Barcelona, Spain (Online). Association for Computational Linguistics.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014)*, pages 4585–4592, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Henry Elder. 2020. ADAPT at SR'20: How preprocessing and data augmentation help to improve surface realization. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 30–34, Barcelona, Spain (Online). Association for Computational Linguistics.

Farhood Farahnak, Laya Rafiee, Leila Kosseim, and Thomas Fevens. 2020. Surface realization using pretrained language models. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 57–63, Barcelona, Spain (Online). Association for Computational Linguistics.

Hamza Harkous, Isabel Groves, and Amir Saffari. 2020. Have your text and use it too! end-to-end neural data-to-text generation with semantic fidelity. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2410–2424, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1693–1701, Cambridge, MA, USA. MIT Press.

Eduard Hovy, Gertjan van Noord, Guenter Neumann, and John Bateman. 1996. Language generation. *Survey of the State of the Art in Human Language Technology*, pages 131–146.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

Samuel Läubli, Rico Sennrich, and Martin Volk. 2018. Has machine translation achieved human parity? a case for document-level evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4791–4796, Brussels, Belgium. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*, Ernest N. Morial Convention Center, New Orleans.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, (ICLR 2017), Conference Track Proceedings*, Toulon, France. OpenReview.net.

Simon Mille, Anja Belz, Bernd Bohnet, Yvette Graham, and Leo Wanner. 2019. Proceedings of the 2nd Workshop on Multilingual Surface Realisation (MSR 2019). Hong Kong, China. Association for Computational Linguistics.

Simon Mille, Anja Belz, Bernd Bohnet, Emily Pitler, and Leo Wanner. 2018. Proceedings of the First Workshop on Multilingual Surface Realisation. Melbourne, Australia. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Gábor Recski, Ádám Kovács, Kinga Gémes, Judit Ács, and Andras Kornai. 2020. BME-TUW at SR'20: Lexical grammar induction for surface realization. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 21–29, Barcelona, Spain (Online). Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building Natural Language Generation Systems*. Natural Language Processing. Cambridge University Press, New York, NY, USA.

Xiang Yu, Simon Tannert, Ngoc Thang Vu, and Jonas Kuhn. 2020. IMSurReal too: IMS in the surface realization shared task 2020. In *Proceedings of the Third Workshop on Multilingual Surface Realisation*, pages 35–41, Barcelona, Spain (Online). Association for Computational Linguistics.

A Detailed Analysis

A.1 Domain Adaptation

Figure 3 compares the BLEU scores of training models for the deep track on the EWT dataset with and without pre-training on $500K$ synthetic samples with different training epochs. As the figure shows, for the pre-trained model, the domain adaptation phase is almost completed after the first epoch while the non-pre-trained model continues to improve even after 10 epochs.

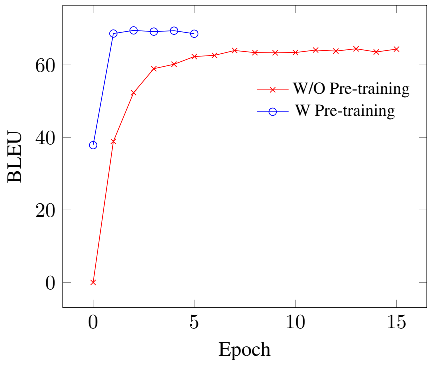

Figure 3: The BLEU score on the EWT test set for all epochs when training the models for the deep track, with and without pre-training on $500K$ synthetic samples.

A.2 Size of Training Data

Table 2 shows that pre-training on synthetic data before fine-tuning on manually labeled data can improve the overall performance of the model. In order to better understand the importance of the size of manually labeled training data, we limited its size and fine-tuned the model on different sizes of manually labeled training data. Table 4 shows the performances of the model after fine-tuning on different subsets of the EWT dataset. As Table 4 shows, fine-tuning solely on $1K$ samples can achieve better performance compared to no pre-training and using the full EWT dataset (last row of Table 4). However, by increasing the number of training samples (from $1K$ to $12.5K$), we can achieve a higher performance when pre-training. This indicates that even though the pre-training phase is helpful for the task, it is not sufficient to replace the manually labeled training data altogether.

# synthetic samples	# manually labeled samples	BLEU	NIST	DIST
$500K$	0	37.87	8.09	61.04
$500K$	1K	64.54	11.88	78.50
$500K$	2K	65.70	12.00	78.93
$500K$	5K	67.35	12.33	80.76
$500K$	10K	68.79	12.43	81.80
$500K$	12.5K	69.52	12.54	82.43
0	12.5K	64.31	11.64	77.80

Table 4: Comparison of the performance of the encoder-decoder model using different sizes of training data for the fine-tuning on a model pre-trained with $500K$ synthetic samples.

A.3 Error Analysis

We manually inspected the errors generated by our models. While a few generated sentences did contained true errors, most can be regarded as correct alternatives to the ground truth. Table 5 shows a few examples. One common correct alternative was related to the generation of contractions as in Ex. 1. This type of error occurs because the MSR input structure of the token to generate (*it*) does not contain any feature that give the model a clue as to whether the token should be contracted or not. In Ex. 2, the model failed to generate the expected punctuation in the deep track, yet the generated sentence is a correct alternative to the ground truth. In Ex. 3, the word order in the generated outputs is not identical to the ground truth; however, they are grammatically correct and convey the same meaning. In Ex. 4, the output of the shallow model is indeed a true error as it is not grammatically correct; however, the deep model generated a grammatically correct output but again it is not identical to the expected output. Finally, in Ex. 5 and 6, show examples of correct alternative to number formatting compared to the ground truth.

Ex. 1	Ground Truth	i 'll post highlights . . .	Ex. 2	Ground Truth	two weeks later , and the violence continues .
	Output of Shallow	i will post highlights . . .		Output of Shallow	two weeks later , and the violence continues .
	Output of Deep	i will post highlights . . .		Output of Deep	two weeks later and the violence continues .
Ex. 3	Ground Truth	they own blogger , of course .	Ex. 4	Ground Truth	we have this report ?
	Output of Shallow	of course , they own blogger .		Output of Shallow	have we this report ?
	Output of Deep	of course they own blogger .		Output of Deep	do we have this report ?
Ex. 5	Ground Truth	compensation : $ 60000 - 70000	Ex. 6	Ground Truth	. . . said that there was a 10 to 50 % chance . . .
	Output of Shallow	compensation : $ 60,000 - 70,000		Output of Shallow	. . . said that there was a 10 to 50 % chance . . .
	Output of Deep	compensation : $ 60000 - 70000		Output of Deep	. . . said there was a 10 - 50 % chance . . .

Table 5: Sample errors generated by the shallow and deep models pre-trained on $500K$ synthetic data and fine-tuned on the EWT dataset.