Ensemble-based Fine-Tuning Strategy for Temporal Relation Extraction from the Clinical Narrative

Lijing Wang[1], **Timothy Miller**[1], **Steven Bethard**[2], **Guergana Savova**[1]
[1]Boston Children's Hospital and Harvard Medical School
[2]University of Arizona
[1]{first.last}@childrens.harvard.edu
[2]bethard@email.arizona.edu

Abstract

In this paper, we investigate ensemble methods for fine-tuning transformer-based pretrained models for clinical natural language processing tasks, specifically temporal relation extraction from the clinical narrative. Our experimental results on the THYME data show that ensembling as a fine-tuning strategy can further boost model performance over single learners optimized for hyperparameters. Dynamic snapshot ensembling is particularly beneficial as it fine-tunes a wide array of parameters and results in a 2.8% absolute improvement in F1 over the base single learner.

1 Introduction

The clinical narrative in electronic medical records (EMRs) can provide critical information for improving quality of care, patient outcomes, and safety. Extracting information from EMRs has been an active area of research in recent years due to the advances in natural language processing (NLP) techniques. As transformer-based neural language models, such as Bidirectional Encodings Representations from Transformers (BERT) (Devlin et al., 2019), have achieved state-of-the-art performance for a variety of NLP tasks they have gained increased prominence in clinical NLP.

However, in the clinical domain, data is often sparsely labeled and not shareable as it is guarded by patient confidentiality provisions. Building large transformer-based models from scratch using such data is thus often infeasible. A common approach has been to take models pretrained on large general domain corpora, and continue pretraining them on clinical corpora to derive domain-specific language models (Lee et al., 2020; Alsentzer et al., 2019; Beltagy et al., 2019; Lin et al., 2021).

The weights of pretrained models are adjusted for a specific clinical NLP task through the process of *fine-tuning*. This process often involves searching for optimal hyperparameters while continuing to train the pretrained model on a domain-specific dataset. The search is challenging due to the high dimensionality of the search space, which includes random seed, initial learning rate, batch size, etc. Given the limited computing resources available in practice, only a small number of values for each hyperparameter can be explored, and often only a subset of hyperparameters can be fine-tuned. Are we able to retain the benefits from the existing search efforts and to further improve model performance for the same task or new tasks without too much extra effort? Ensemble methods have been successful in boosting predictive performance of single learners (Wang et al., 2003; CireşAn et al., 2012; Xie et al., 2013) and thus are promising. In this paper, we will investigate ensemble-based fine-tuning methods to answer this question.

Another downside of the limited search capability is that some hyperparameters are unexplored in past efforts. For example, learning rate schedules have rarely been explored in previous efforts of fine-tuning. One promising approach is training with cyclical learning rates (e.g., cosine annealing learning rate and slanted triangular learning rate), which have been shown to achieve improved classification accuracy in fewer iterations (Loshchilov and Hutter, 2016; Smith, 2017). We will explore the impact of cyclical learning rates in fine-tuning methods in the context of an ensemble algorithm.

Major contributions: In this work, (1) we use ensembles to investigate the impact of various hyperparameters for fine-tuning pretrained transformer-based models for the clinical domain by focusing on one critical task – temporal relation extraction; (2) we conduct comprehensive experiments and the empirical findings show that training epoch, random initialization, and data order have potentially significant influence; (3) we explore multiple hyperparameters in a single framework with the aim of building computationally efficient fine-tuning strategies to boost model performance on top of

Proceedings of the 4th Clinical Natural Language Processing Workshop, pages 103 - 108
July 14, 2022 ©2022 Association for Computational Linguistics

any given base setting.

2 Temporal Relation Extraction in Clinical Narratives

We explore the ensemble-based fine-tuning methods within the context of temporal relation extraction from the EMR clinical narrative. Temporal relation extraction and reasoning in the clinical domain continues to be a primary area of interest due to the potential impact on disease understanding and, ultimately, patient care. A significant body of text available for this purpose is the THYME (Temporal Histories of Your Medical Events) corpus (Styler IV et al., 2014), consisting of 594 de-identified clinical and pathology notes on colon cancer patients and 600 radiology, oncology and clinical notes on brain cancer patients, all from the EMR of a leading US medical center. This dataset has previously undergone a variety of annotation efforts, most notably temporal annotation (Styler IV et al., 2014). It has been part of several SemEval shared tasks such as Clinical TempEval (Bethard et al., 2017) where state-of-the-art results have been established. We use the THYME++ version of the corpus and the train/dev/test splits as described by Wright-Bettner et al. (2020).

3 Ensemble-based Fine-Tuning and Experimental Setup

Our intuition behind using ensembles for fine-tuning is to leverage models from local optima to obtain greater coverage of the feature space, and get consensus for the predictions so that the ensemble learner can reduce the overall risk of making a poor selection. In this section, we first describe our setting and implementation of a base model based on the state-of-the-art setting described by Lin et al. (2021). Then we discuss fine-tuning several hyperparameters during training and their potential impact on model performance. Based on these discussions, we then introduce the bagging ensemble method (Breiman, 1996) and the dynamic snapshot ensemble method (Wang et al., 2020) and apply them to the fine-tuning process.

3.1 Base setting and implementation

To set up an ensemble learning method, we first need to set up a base setting as a starting point. Based on the results and discussions of Lin et al. (2021), we choose `PubmedBERTbase-MimicBig-EntityBERT`[1] as our pretrained model. The fine-tuning setting in that work includes random seed 42, batch size 32, epoch number 3, learning rate 4e-5, learning rate scheduler *linear*, max sequence length 100, and gradient accumulation steps 2. We adopt the same setting in our base implementation. We use an NVIDIA Titan RTX GPU cluster of 7 nodes for fine-tuning experiments through HuggingFace's Transformer API (Wolf et al., 2020) version 4.13.0. We leverage the `run_glue.py` pytorch version as our fine-tuning script. Unless specified, default settings are used in our experiments. Due to differences in the fine-tuning script and some missing settings, we were unable to reproduce the exact scores reported in Lin et al. (2021). Results with our implementation are reported as BASE. We use our implementation as the starting point to conduct the ensemble experiment and compare ensemble results with BASE.

3.2 Hyperparameters in fine-tuning

There are more than a hundred hyperparameters in the fine-tuning process. Among those hyperparameters, not every one has a major impact on model performance. Some of them are preset with default values that have been shown to be robust in empirical experiments, such as the default values of β_1, β_2, and ϵ for AdamW optimizer. In our work, we investigate several hyperparameters which potentially have high impact on model performance. We apply ensemble learning on the following hyperparameters to reduce the variance of predictions and reduce generalization error:

Random seed is set at the beginning of training. It impacts the initialization of models and trainers, as well as the convergence of scholastic learning algorithms. We run base fine-tuning 5 times but with 5 random seed values (42, 52, 62, 72, 82).

Learning rate scheduler is the scheduling algorithm for changing the learning rate during training. In the previous fine-tuning works, the *linear* scheduler is used by default. We run base fine-tuning with 3 different learning rate schedulers: *linear*, *cosine with restarts*, and *polynomial*.

Epoch number is the number of passes over the data that the training process takes. A small epoch number may lead to underfitting while a large epoch number tends to cause overfitting to

[1] `https://physionet.org/content/entity-bert/1.0.0/`

the domain-specific training data. We run the base fine-tuning with 5 epoch numbers (3, 6, 9, 12, 15).

Pretrained model is the model checkpoint from which fine-tuning begins. The PubMedBERT model (Gu et al., 2021) has been shown to outperform other BERT-based models for temporal relation extraction in clinical narratives (Lin et al., 2021). In our experiments, we leverage the three PubMedBERT models released by Lin et al. (2021): `PubmedBERTbase-MimicBig-EntityBERT`, `PubmedBERTbase-MimicSmall-EntityBERT`, and `PubmedBERTbase-MimicBig-RandMask`.

Random shuffling of training and validation data can avoid selecting models that overfit to a single validation set during fine-tuning. In contrast to traditional random shuffling of training instances during training, the random shuffling in this work refers to mixing training and validation datasets and then resampling train/validation datasets with the same size and class distribution from the mix pool. We generate 5 different samplings of splits using random seeds (42, 52, 62, 72, 82). We then run base fine-tuning 5 times with different samplings.

3.3 Bagging ensemble

Bagging ensemble is the simple and straightforward thus is commonly used in various tasks. Component learners are trained independently in parallel and are combined following some kind of combination method. We leverage bagging ensemble and use majority voting for generating ensemble predictions on each hyperparameter variable. For example, for the random seed variable, we combine predictions from 5 fine-tuned models trained with different random seeds using majority voting, denoted as Seed-ENS. We report the ensemble performance regarding each hyperparameter variable in Table 1 together with BASE.

3.4 Dynamic snapshot ensemble

We also explore dynamic snapshot ensembles first proposed in (Wang et al., 2020), which we call DynSnap-ENS in this paper. The DynSnap-ENS framework allows a pretrained model to be fine-tuned multiple times (i.e., multiple training runs) sequentially with different random seeds and data samplings of train/validation splits. It uses a cyclic annealing schedule and cyclic snapshot strategy to periodically save the best model during each training run. After each training run, a dynamic pruning algorithm is applied to select a few single learners from the saved ones which can lead to better performance of the ensemble learner with theoretical guarantees. The sequential training runs stop when the accumulated number of selected single learners reaches a preset ensemble size. The total amount of training runs is a dynamic value rather than a preset value, which is determined by the snapshot strategy and pruning factor during the sequential training. Take Figure 1 as an example. The preset ensemble size is 5, and training epoch is 15. Training run1 is set with random seed 42 and a data split. After the training, top 4 models are saved based on validation accuracy, and among those 2 models are selected as ensemble components after pruning. Since 2 is smaller than 5, training run2 is triggered with random seed 52 and another data split. This process will repeat until the accumulated number of ensemble components reaches the ensemble size. More details of the learning algorithm can be found in the original paper.

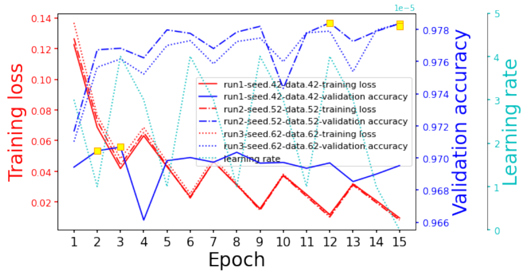

Figure 1: Training history of DynSnap-ENS on learning rate, training loss, and validation accuracy along epochs. Ensemble size is 5. The sequential training runs are run1-run2-run3. The selected single learners are highlighted with yellow squares.

We are the first to apply DynSnap-ENS to solve challenges in clinical text classifications. It enables diversity in data and model parameters through a cyclic learning rate, multiple random seeds, epoch numbers, and training and validation datasets. These hyperparameters are explored in one learning framework, which is computationally efficient compared to independent searches for each hyperparameter in Lin et al. (2021).

In our experiments, we implemented DynSnap-ENS on the top of the base fine-tuning script. The ensemble size is set as 5 (equal to the ensemble size of bagging ensemble learners) and majority voting is used to generate ensemble predictions. We reuse base fine-tuning settings except that we set *cosine with restarts* as the learning rate scheduler and set the learning rate to restart every 3 epochs

	OVERLAP			**CONTAINS-1**			**CONTAINS**			**BEFORE-1**			**BEFORE**		
Method	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BASE	0.611	**0.482**	0.539	0.749	0.758	0.754	0.775	0.777	0.776	0.51	**0.428**	0.465	0.537	0.416	0.469
Seed-ENS	0.672	0.46	0.546	0.753	0.757	0.755	0.785	**0.79**	0.788	0.562	0.404	0.47	0.57	0.411	0.477
LRScheduler-ENS	0.652	0.48	0.553	0.741	0.758	0.749	0.789	0.781	0.785	0.535	0.406	0.462	0.568	0.396	0.467
Epoch-ENS	0.681	0.471	0.556	**0.774**	0.765	**0.769**	0.807	0.779	0.793	**0.599**	0.376	0.462	0.627	0.379	0.472
PretrainedModel-ENS	0.676	0.458	0.546	0.735	0.769	0.752	0.786	0.788	0.787	0.536	0.42	**0.471**	0.564	0.408	0.473
DataShuffle-ENS	**0.711**	0.458	**0.557**	0.737	**0.771**	0.754	0.806	0.788	**0.797**	0.586	0.384	0.464	0.617	**0.429**	**0.506**
DynSnap-ENS	0.695	0.464	**0.557**	0.769	0.762	0.766	**0.816**	0.778	0.796	0.579	0.381	0.459	**0.636**	0.404	0.494

	NOTED-ON-1			**BEGINS-ON**			**NOTED-ON**			**ENDS-ON**			**OVERALL**		
Method	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
BASE	0.739	0.824	0.779	0.637	0.581	0.608	0.706	0.55	0.618	0.773	0.574	0.659	0.671	0.599	0.633
Seed-ENS	0.766	0.809	0.787	0.705	0.537	0.61	**0.794**	0.55	0.65	0.799	0.602	0.687	0.712	0.591	0.646
LRScheduler-ENS	0.765	0.81	0.787	0.669	0.569	0.615	0.792	0.543	0.644	0.763	0.582	0.66	0.697	0.592	0.640
Epoch-ENS	**0.771**	0.816	0.793	**0.771**	0.569	0.655	0.782	0.564	0.656	0.807	**0.635**	0.711	0.732	0.596	0.657
PretrainedModel-ENS	0.769	0.801	0.784	0.664	0.531	0.59	0.777	0.521	0.624	0.812	0.602	0.692	0.702	0.589	0.640
DataShuffle-ENS	0.758	**0.832**	0.793	0.682	0.562	0.616	0.758	0.536	0.628	**0.854**	0.553	0.672	0.723	0.590	0.650
DynSnap-ENS	0.768	0.822	**0.794**	0.726	**0.613**	**0.664**	0.777	**0.571**	**0.658**	0.831	0.623	**0.712**	**0.733**	**0.602**	**0.661**

Table 1: Ensemble model performance on THYME test colon data. NONE - no relation, CONTAINS-1 - arg 2 contains arg 1, CONTAINS - arg 1 contains arg2, BEFORE-1 - arg 2 before arg 1, BEFORE - arg 1 before arg 2, NOTED-ON-1 - arg 2 noted on arg 1, BEGINS-ON - arg 1 begins on arg 2, NOTED-ON - arg 1 noted on arg 2, ENDS-ON - arg 1 ends on arg 2. NONE scores are omitted from the table and the OVERALL is the macro average score excluding NONE.

which, based on the base setting, allows the model to converge to a reasonable state before each restart. The total number of epochs for each training run is 15 and we save the top 4 models for pruning based on validation accuracy. The random seeds and shuffling datasets for the sequential training runs are the same with the 5 options described in Section 3.2. The logic behind the above settings is to retain the benefits from the base fine-tuning settings as much as possible. Codes and settings to reproduce the results are available here[2].

4 Results and Discussion

We show model performance in Table 1. Compared with BASE, all ensemble methods boost the overall F1 score, with DynSnap-ENS achieving the highest improvement, 2.8% absolute. The improvement is mainly due to the increase in precision, 6.2% absolute. This complies with the theoretical findings in Wang et al. (2020) that ensemble can improve prediction accuracy (i.e. precision). However, there is no proof that ensembling can improve recall.

Among the bagging ensembles, diversity in epoch number (Epoch-ENS) leads to the largest improvement, 2.4% absolute. Diversity in data order (DataShuffle-ENS) and random seeds (Seed-ENS) achieve the next best improvement, 1.7% and 1.3% absolute, while diversity in learning rate schedulers (LRScheduler-ENS) and PubMedBERT variants (PretrainedModel-ENS) obtain the least improvement, 0.7% absolute. In general, we see that selecting a single model is a riskier choice than ensembling several models when trying to avoid overfitting or underfitting the training data.

However, all sources of diversity are not equal, with the diversity from different epochs of a training run being most helpful, and diversity of learning rate schedulers and diversity of PubMedBERT variants helping little. A possible reason is that both LRScheduler-ENS and PretrainedModel-ENS have only 3 components while the other ensemble learners have 5 components, as Wang et al. (2020) proved that a better precision can be achieved if more component learners are combined. However, that would not explain the superiority of Epoch-ENS to DataShuffle-ENS and Seed-ENS, and an improvement of the ensemble's performance is not guaranteed if many poor learners are combined. DynSnap-ENS outperforms all the other ensemble learners, likely because it takes advantage of all the individual types of diversity: data, model parameters, epochs, and learning rate. Figure 1 presents the training history on learning rate, training loss, and validation accuracy along epochs. We can observe that learning behavior changes a lot with respect to each source of diversity. DynSnap-ENS combines those sources in a computationally effi-

[2] https://github.com/christa60/transformer-dynamic-snapshot-ensemble

cient way and selects top single learners (marked in yellow squares) from a more diversified pool to guarantee an improvement in the final ensemble learner.

5 Conclusion

We investigated ensemble methods in fine-tuning transformer-based pretrained models for clinical NLP tasks, specifically temporal relation extraction from the clinical narrative. Our experimental results on the THYME++ data showed that ensembling can further boost performance, and that dynamic snapshot ensembling is especially effective. Future works include: 1) investigating the impact of ensemble size in model performance; 2) exploring hyperparameters regarding the snapshot strategy and pruning algorithm; 3) testing the trained ensemble learners on an expanded set of clinical domain tasks.

Acknowledgements

The study was funded by R01LM013486, R01LM10090 and U24CA248010 from the Unites States National Institutes of Health. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Steven Bethard, Guergana Savova, Martha Palmer, and James Pustejovsky. 2017. SemEval-2017 task 12: Clinical TempEval. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 565–572, Vancouver, Canada. Association for Computational Linguistics.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24(2):123–140.

Dan CireşAn, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. 2012. Multi-column deep neural network for traffic sign classification. *Neural networks*, 32:333–338.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Trans. Comput. Healthcare*, 3(1).

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2021. EntityBERT: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Leslie N Smith. 2017. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE.

William F. Styler IV, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet C de Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, and James Pustejovsky. 2014. Temporal annotation in the clinical domain. *Transactions of the Association for Computational Linguistics*, 2:143–154.

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. 2003. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235. AcM.

Lijing Wang, Dipanjan Ghosh, Maria Gonzalez Diaz, Ahmed Farahat, Mahbubul Alam, Chetan Gupta, Jiangzhuo Chen, and Madhav Marathe. 2020. Wisdom of the ensemble: Improving consistency of deep learning models. *Advances in Neural Information Processing Systems*, 33:19750–19761.

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, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Kristin Wright-Bettner, Chen Lin, Timothy Miller, Steven Bethard, Dmitriy Dligach, Martha Palmer, James H. Martin, and Guergana Savova. 2020. Defining and learning refined temporal relations in the clinical narrative. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 104–114, Online. Association for Computational Linguistics.

Jingjing Xie, Bing Xu, and Zhang Chuang. 2013. Horizontal and vertical ensemble with deep representation for classification. *arXiv preprint arXiv:1306.2759*.