Hyperparameter Optimization in NLP Architectures

Noureddine Ettaik[a], Ben Lahmar El Habib[b]
Laboratory for Information Processing and Modeling, Hassan II University of Casablanca, Faculty of Sciences Ben M'sik, BP 7955, Sidi Othman, Casablanca, Morocco

Keywords: Natural Language processing, Hyperparameter optimization, Survey

Abstract: Hyperparameter optimization (HPO) is an essential part of setting up efficient machine learning models dealing with natural language processing (NLP) tasks, especially with the recent NLP breakthroughs. In this paper, we explore the problem of HPO through a survey conducted on a selected number of academic publications in NLP by studying the strategy used for the optimization of their hyperparameters and by investigating their common traits. We then lay out some recommendations for good practice in NLP HPO.

1 INTRODUCTION

As data grows in size and complexity, algorithms processing it are getting more complex, especially when dealing with natural language processing NLP tasks. A significant component of this complexity stems from the optimization process of hyperparameters.

Hyperparameter optimization is mainly done manually (Hinton 2012) or by testing a predefined set of hyperparameters in a grid (Pedregosa et al. 2011), or by exploring the search space and randomly selecting the optimal hyperparameters (Bergstra and Bengio 2012). More sophisticated techniques have made their appearance (Claesen and De Moor 2015) and were applied to HPO like the Bayesian optimization (Swersky, Snoek, and Adams 2013; Zhang et al. 2016; Klein et al. 2017; Bergstra et al. 2015; Feurer et al. 2015), the evolutionary algorithms (Young et al. 2015), heuristics inspired algorithms like simulated annealing, particle swarm optimization (Ye 2017) and ant colony optimization (Costa and Rodrigues 2018).

Deep learning models share a multitude of their models' hyperparameters, but when dealing with a specific task like in NLP with specific architectures like RNNs and LSTMs, or more recent transformer-based ones, generalizations cannot be easily made, and the optimization becomes more task-specific. In this context, hyperparameters tend to behave according to the task at hand and the probability of generalization among models dealing with NLP is higher in the same architecture.

In this paper, we inquire into the use of hyperparameter optimization in natural language processing by investigating a group of papers in the domain, we try to spot the techniques used to optimize these hyperparameters as well as the tools used to tune them, we also try to find any pattern related to the most important and useful hyperparameters and set them apart from the less useful ones. We finally set a list of recommendations regarding the use of hyperparameter optimization in the context of NLP.

2 STUDIES OVERVIEW

The studies selected for this paper are a result of research in academic databases based on an adapted search equation, a study selection was applied on the resulting papers (inclusion and exclusion criteria). They are highly cited papers in the domain of NLP dealing with hyperparameter optimization and published in highly respected journals and conferences in the domain of artificial intelligence. We describe below the major highlights of HPO in NLP displayed in each study.

In (Melis, Dyer, and Blunsom 2017), the focus is on language modelling architectures like LSTMs, RHN (Recurrent highway network) (Zilly et al. 2017)

[a] https://orcid.org/0000-0001-8523-1227
[b] https://orcid.org/0000-0001-7098-4621

Ettaik, N. and El Habib, B.
Hyperparameter Optimization in NLP Architectures.
DOI: 10.5220/0010736600003101
In *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning (BML 2021)*, pages 466-470
ISBN: 978-989-758-559-3

and NAS (Neural architecture search) (Zoph and Le 2017), the HPO is performed on Google vizier (Golovin et al. 2017), which is a black box hyperparameter optimization tool. Based on the results of this Automl tuner and the validation loss, the authors plotted these results against each other, which showed that 15 %to 25% of the hyperparameter space is promising, a comparable grid search yielding the same results, would have taken too much time and trials as the authors stated.

In (Aghaebrahimian and Cieliebak 2019), an ad hoc search was used to tune the hyperparameters of a model over a limited budget for a text classification task. Certain hyperparameters had an impact on the model, while others had none.

For the same task, we can see that the authors in (Tellez et al. 2017), tackle the task of text categorization in a supervised learning layout, the process of optimization was carried out by first performing a combination of a random search and a hill-climbing algorithm, in other words, the RS spots randomly some promising areas of the hyperparameter space and then the hill-climbing algorithm is initialized using those promising points by greedily updating them till no improvement is possible.

The task of linguistic sequence tagging is investigated in (Reimers and Gurevych 2017) using deep LSTM-Networks, the study shows that hyperparameters have different impacts on the model. For example, the choice of embeddings and the dropout mechanism have a considerable leverage on the result. Also, the hyperparameters tend to influence each other, which means if an initial search yielded an excellent performance, the global optima might be hidden down the line of further HP combinations.

In (Dernoncourt and Lee 2016), the task at hand is based on dialog act classification, more precisely assigning a dialog act to each utterance, and the model is based on an ANN architecture. The hyperparameters are optimized using Gaussian process GP (a Bayesian optimization inspired approach), which proved to be four times better than the uninformed random search. That GP search result is highly impacted by the initial random points.

The task at hand In (Wang et al. 2015) is classification and question answering, and the optimization of hyperparameters is done based on Bayesian optimization but with a multi-stage approach allowing for the HPO to take place in successive stages in a sequential fashion while increasing the amount of data being trained. The optimization starts with a small amount of data being trained in the early stages and keeps improving according to the promising candidate hyperparameters, a full Bayesian optimization is applied based on these candidates to allow for a convergence based on a better prior knowledge.

The transformer architecture with self-attention introduced in (Vaswani et al. 2017) experimented on machine translation, and the hyperparameters were set after multiple experimentations on different values (No mention of a particular technique of optimization), especially the optimizer and its parameters, the regularization (residual dropout, attention drop out and label smoothing) and the learning rate which underwent an increase according to a formula with warmup step of 4000.

In (Devlin et al. 2019), another transformer-based architecture paper with an outcome of a state of the art pre-trained model (BERT), the tasks it tackles are question answering and language understanding. For this type of model, transfer learning can be applied with minor hyperparameter tuning; in fact; the optimization can take place using the majority of the pre-trained model hyperparameter values and use a recommended range of values for the remaining hyperparameters (dropout, batch size, learning rate and epochs) even with tasks different than the original tasks of the model.

3 HYPERPARAMETERS PATTERN IN NLP

In table 1, we lay down the pattern we detected in the studies, which showcases the hyperparameter optimization HPO used, and the NLP specified task(s).

Table 1: HPO techniques in NLP and tasks by study.

Study	Technique	NLP task
(Dernoncourt and Lee 2016)	Bayesian opt by gaussian process	Dialog act classification
(Caselles-Dupré et al. 2018)	Manual search	Natural Language Processing tasks and Recommendation tasks (word to vec: embeddings)
(Tellez et al. 2017)	Random Search then Hill-Climbing	Text classification (topic and polarity classification, spam detection, user profiling and authorship attribution)

(Wang et al. 2015)	Multi-stage Bayesian optimization	Sentiment classification and QA tasks
(Aghaebrahimian and Cieliebak 2019)	Ad hoc grid search	Multi-label text classification task
(Reimers and Gurevych 2017)	NA	Linguistic sequence tagging (POS, Chunking, NER, Entity Recognition, and Event Detection)
(Devlin et al. 2019)	NA	Question Answering; natural language understanding
(Vaswani et al. 2017)	HP set with expertise and optimization with a formula (for LR)	Machine translation

4 HPO STRATEGIES

The strategy of hyperparameter optimization adopted depends on the task at hand, the available resources and the dataset complexity. The Bayesian optimization proves to be efficient but is very expensive; thus performing a multi-stage BO can yield better and faster results (Wang et al., 2015). Its variant; the gaussian process; proves to be faster than random search in dialog act classification (Dernoncourt & Lee, 2016), when using simple heuristics may not find optimal hyperparameters well. When dealing with multilabel text classification tasks, the degree of impact of the HP is not the same, and some have no impact at all on the model; for example in (Aghaebrahimian & Cieliebak, 2019; Reimers & Gurevych, 2017), the combination of glove embedding type, Sigmoid for last layer, bi GRU deep architecture, Nadam as optimizer, pooling with both max and average concatenated, normalization for Gradient control and dropout set to variational proved to be the optimal set of HP for the task.

In tasks related to linguistic sequence tagging, the combination of the optimizer Nadam with a gradient normalization threshold of 1, a variational dropout, recurrent units of LSTM layer and a CRF classifier (Reimers & Gurevych, 2017). Relying on full Automl search may be counterproductive as an HPO strategy since it could be greedy. A simple combination with a manual approach (Melis et al., 2017), for example, a plot between promising hyperparameters and validation loss could produce optimal results.

For machine translation tasks, and under a transformer architecture (Vaswani et al., 2017) recommend choosing specific hyperparameter values, namely Adam as optimizer, a learning rate that is variant according to the warm-up steps, which start at 4000, a residual dropout rate of 0.1 and a label smoothing of 0.1. in the same transformer environment but with a bidirectional variant and a different task based on Question Answering and natural language understanding (Devlin et al., 2019) recommend setting dropout probability to 0.1, batch size to16 or 32, learning rate (Adam) to 5e-5, 3e-5 or 2e-5 and epochs to 2, 3 or 4.

5 ANALYSIS AND DISCUSSION

NLP tasks are becoming more and more demanding, and the need for optimization grows more important than ever. Meanwhile, hyperparameter optimization algorithms and techniques have recently proven to be very efficient, especially with the rise of automated machine learning applications, and if used efficiently, could improve the state-of-the-art models and bypass the budget and time constraints.

Not all hyperparameters have the same impact on the models, and the nature of NLP tasks favours parameters like the word embeddings type, the dropout rate and the optimizer, which are more impactful than others, thus making the tuning of the latter useless in most cases.

HPO is becoming as compulsory as choosing the model itself for NLP tasks. Researchers and practitioners alike should invest more time and effort to optimize their models using the available techniques.

A combination of automatic HPO based mostly on a sophisticated approach like the Bayesian technique coupled with a rule of thumb like a plot comparison between the initial HPO and the loss, a warm start, or an early stopping is the most efficient since a complete search is always greedy in resources.

The availability of pre-trained models like BERT (Devlin et al., 2019), ELMO (Peters et al., 2018) and others allows for a minimal hyperparameter optimization that doesn't involve a huge search space and can be applied across a wide array of NLP tasks.

Applying different HPO techniques on NLP models may be time-consuming or expensive for some tasks, especially with big datasets, but their use proves to be effective when the sweet spot of optimal values is reached, especially with the use of pre-trained models since they are highly effective in the field of NLP, and the tasks in the domain are recurrent and repetitive and deal with a homogenous entity which is language. Meaning that the hyperparameters that are optimal in a big model will very likely yield the same results in smaller tasks with the same nature.

6 CONCLUSIONS

This paper provides an insight into good practices in hyperparameter optimization in natural language processing related tasks. We found out that there are common traits in the optimization process of hyperparameters and that some particular HPO techniques work well with certain tasks. Also, the values reported in this paper from certain studies can be reproduced in similar tasks. The recent developments in transformer architectures, have paved the way for optimal models down the line by means of transfer learning, which benefits ultimately the hyperparameter optimization in NLP.

REFERENCES

Moore, R., Lopes, J., 1999. Paper templates. In *TEMPLATE'06, 1st International Conference on Template Production*. SCITEPRESS.

Smith, J., 1998. *The book*, The publishing company. London, 2^{nd} edition.

Aghaebrahimian, Ahmad, and Mark Cieliebak. 2019. "Hyperparameter Tuning for Deep Learning in Natural Language Processing," 7.

Bergstra, James, and Yoshua Bengio. 2012. "Random Search for Hyper-Parameter Optimization," 25.

Bergstra, James, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. 2015. "Hyperopt: A Python Library for Model Selection and Hyperparameter Optimization." Computational Science & Discovery 8 (1): 014008. https://doi.org/10.1088/1749-4699/8/1/014008.

Caselles-Dupré, Hugo, Florian Lesaint, and Jimena Royo-Letelier. 2018. "Word2Vec Applied to Recommendation: Hyperparameters Matter." ArXiv:1804.04212 [Cs, Stat], August. http://arxiv.org/abs/1804.04212.

Claesen, Marc, and Bart De Moor. 2015. "Hyperparameter Search in Machine Learning." ArXiv:1502.02127 [Cs, Stat], April. http://arxiv.org/abs/1502.02127.

Costa, Victor O., and Cesar R. Rodrigues. 2018. "Hierarchical Ant Colony for Simultaneous Classifier Selection and Hyperparameter Optimization." In 2018 IEEE Congress on Evolutionary Computation (CEC), 1–8. Rio de Janeiro: IEEE. https://doi.org/10.1109/CEC.2018.8477834.

Dernoncourt, Franck, and Ji Young Lee. 2016. "Optimizing Neural Network Hyperparameters with Gaussian Processes for Dialog Act Classification." In 2016 IEEE Spoken Language Technology Workshop (SLT), 406–13. San Diego, CA: IEEE. https://doi.org/10.1109/SLT.2016.7846296.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." ArXiv:1810.04805 [Cs], May. http://arxiv.org/abs/1810.04805.

Feurer, Matthias, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. 2015. "Efficient and Robust Automated Machine Learning," 9.

Golovin, Daniel, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. 2017. "Google Vizier: A Service for Black-Box Optimization." In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17, 1487–95. Halifax, NS, Canada: ACM Press. https://doi.org/10.1145/3097983.3098043.

Hinton, Geoffrey E. 2012. "A Practical Guide to Training Restricted Boltzmann Machines." In Neural Networks: Tricks of the Trade, edited by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller, 7700:599–619. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-35289-8_32.

Klein, Aaron, Stefan Falkner, Simon Bartels, Philipp Hennig, and Frank Hutter. 2017. "Fast Bayesian Hyperparameter Optimization on Large Datasets." Electronic Journal of Statistics 11 (2): 4945–68. https://doi.org/10.1214/17-EJS1335SI.

Komninos, Alexandros, and Suresh Manandhar. 2016. "Dependency Based Embeddings for Sentence Classification Tasks." In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 1490–1500. San Diego, California: Association for Computational Linguistics. https://doi.org/10.18653/v1/N16-1175.

Melis, Gábor, Chris Dyer, and Phil Blunsom. 2017. "On the State of the Art of Evaluation in Neural Language Models." ArXiv:1707.05589 [Cs], November. http://arxiv.org/abs/1707.05589.

Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." MACHINE LEARNING IN PYTHON, 6.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. "Deep Contextualized Word Representations." ArXiv:1802.05365 [Cs], March. http://arxiv.org/abs/1802.05365.

Reimers, Nils, and Iryna Gurevych. 2017. "Optimal Hyperparameters for Deep LSTM-Networks for

Sequence Labeling Tasks." ArXiv:1707.06799 [Cs], August. http://arxiv.org/abs/1707.06799.

Rijn, J. N. van, and F. Hutter. 2018. "Hyperparameter Importance Across Datasets." Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18, 2367–76. https://doi.org/10.1145/3219819.3220058.

Swersky, Kevin, Jasper Snoek, and Ryan P Adams. 2013. "Multi-Task Bayesian Optimization," 9.

Tellez, Eric S., Daniela Moctezuma, Sabino Miranda-Jímenez, and Mario Graff. 2017. "An Automated Text Categorization Framework Based on Hyperparameter Optimization." ArXiv:1704.01975 [Cs, Stat], September. http://arxiv.org/abs/1704.01975.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. "Attention Is All You Need." ArXiv:1706.03762 [Cs], December. http://arxiv.org/abs/1706.03762.

Wang, Lidan, Minwei Feng, Bowen Zhou, Bing Xiang, and Sridhar Mahadevan. 2015. "Efficient Hyper-Parameter Optimization for NLP Applications." In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2112–17. Lisbon, Portugal: Association for Computational Linguistics. https://doi.org/10.18653/v1/D15-1253.

Wistuba, Martin, Nicolas Schilling, and Lars Schmidt-Thieme. 2015. "Learning Data Set Similarities for Hyperparameter Optimization Initializations." In Metasel@ Pkdd/Ecml, 15–26.

Ye, Fei. 2017. "Particle Swarm Optimization-Based Automatic Parameter Selection for Deep Neural Networks and Its Applications in Large-Scale and High-Dimensional Data." Edited by Wen-Bo Du. PLOS ONE 12 (12): e0188746. https://doi.org/10.1371/journal.pone.0188746.

Young, Steven R., Derek C. Rose, Thomas P. Karnowski, Seung-Hwan Lim, and Robert M. Patton. 2015. "Optimizing Deep Learning Hyper-Parameters through an Evolutionary Algorithm." In Proceedings of the Workshop on Machine Learning in High-Performance Computing Environments - MLHPC '15, 1–5. Austin, Texas: ACM Press. https://doi.org/10.1145/2834892.2834896.

Zhang, Yuyu, Mohammad Taha Bahadori, Hang Su, and Jimeng Sun. 2016. "FLASH: Fast Bayesian Optimization for Data Analytic Pipelines." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16, 2065–74. San Francisco, California, USA: ACM Press. https://doi.org/10.1145/2939672.2939829.

Zilly, Julian Georg, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. 2017. "Recurrent Highway Networks." ArXiv:1607.03474 [Cs], July. http://arxiv.org/abs/1607.03474.

Zoph, Barret, and Quoc V. Le. 2017. "Neural Architecture Search with Reinforcement Learning." ArXiv:1611.01578 [Cs], February. http://arxiv.org/abs/1611.01578.