Discussion: Effective and Interpretable Outcome Prediction by Training Sparse Mixtures of Linear Experts*

Francesco Folino[1], Luigi Pontieri[1,*] and Pietro Sabatino[1]

[1]*Institute for High Performance Computing and Networking (ICAR-CNR), via P. Bucci 8/9C, 87036 Rende (CS), Italy*

Abstract

Process Outcome Prediction entails predicting a discrete property of an unfinished process instance from its partial trace. High-capacity outcome predictors discovered with ensemble and deep learning methods have been shown to achieve top accuracy performances, but they suffer from a lack of transparency. Aligning with recent efforts to learn inherently interpretable outcome predictors, we propose to train a sparse Mixture-of-Experts where both the "gate" and "expert" sub-nets are Logistic Regressors. This ensemble-like model is trained end-to-end while automatically selecting a subset of input features in each sub-net, as an alternative to the common approach of performing a global feature selection step prior to model training. Test results on benchmark logs confirmed the validity and efficacy of this approach.

Keywords

Process Mining, Machine Learning, XAI

1. Introduction

(Process) Outcome Prediction problem [2] refers to the problem of predicting the outcome of an unfinished process instance, based on its associated *prefix trace* (i.e., the partial sequence of events available for the instance). Recently, different supervised learning approaches to this problem were proposed, which allow for discovering an outcome prediction model from labeled traces. Outstanding performances in terms of prediction accuracy have been achieved by big ensembles of decision rules/trees discovered with random forest or gradient boosting algorithms [2], and *Deep Neural Networks* (*DNNs*) [3]. However, the approximation power of these models comes at the cost of an opaque decision logic, which makes them unfit for settings where explainable predictions and interpretable predictors are required. The call for transparent outcome prediction first originated several proposals relying model-agnostic post-hoc explanation methods [4, 5] or explanation-friendly DNN-oriented solutions [6, 7].

Due to widespread concerns on the reliability of attention and post-hoc attribution-based

Proceedings of the 1st International Workshop on Explainable Knowledge Aware Process Intelligence, June 20–22, 2024, Roccella Jonica, Italy

*This paper summarizes results presented at workshop *ML4PM 2023*, associated with conference ICPM 2023, October 23-27, 2023, Rome, Italy, and published in [1].

*Corresponding author.

francesco.folino@icar.cnr.it (F. Folino); luigi.pontieri@icar.cnr.it (L. Pontieri); pietro.sabatino@icar.cnr.it (P. Sabatino)

explanations [8], the discovery of inherently-interpretable outcome predictors [9, 10] was proposed of late. Two alternative kinds of interpretable models, both leveraging *Logistic Regression* (*LR*) models as a building block, were exploited in [9] to discover outcome predictors from flattened log traces: *Logit Leaf Model* (*LLM*), a sort of decision tree where each leaf hosts an LR sub-model; and (ii) and *Generalized Logistic Rule Model* (*GLRM*), where a single LR model is built upon the original feature and novel features, defined as conjunctive rules over subsets of the former and derived via column generation. These kinds of models were both shown to improve plain LR predictors by capturing some non-linear input-output dependencies. An approach leveraging a neural implementation of fuzzy logic, named *FOX*, was proposed in [10] to extract easy-to-interpret IF-THEN outcome-prediction rules, each of which contains a fuzzy set for each (flattened) trace feature and a membership score for each outcome class. Unfortunately, as observed [9] and discussed in Section 3, these LLM-based and GLRM-based methods may return cumbersome models/rules that hardly enable a clear and complete understanding of the predictor's behavior, On the other hand, if the global feature selection and (3-way) feature binning of FOX [10] helps control prediction rules' size, it risks causing some information/accuracy loss.

We next describe an approach to outcome prediction that builds an ensemble of LR models up by training a Mixture of Experts (MoE) neural net. This net consists of multiple "experts" (specialized outcome predictors) and a sparse "gate" module devoted to routing each data instance to one of the experts. For the sake of interpretability, both the gate and experts take the form of LR classifiers (i.e. one-layer neural net with linear activations). Differently from [11], the user is allowed to control the complexity of the model by fixing the maximum number $kTop$ of features that the gate and each expert sub-net can use and the maximum number m of experts. Instead of preliminary performing a global feature selection step as in [9] and [10], first the neural net is trained using all the data features, and then the less important learned parameters are pruned out in a "feature-based" way. This allows different experts to use different subsets of the input features when making their predictions.

2. Proposed Outcome-Prediction Model and Training Algorithm

A *Mixture of Experts* (*MoE*) is a neural net that implements both the gate and the local predictors ("experts") through the composition of smaller interconnected sub-nets. In particular, in classical (dense) MoEs [12], once provided with an input data vector x, the gate is a feed-forward sub-net that computes a vector of (softmax-ed) weights, one for each expert, estimating how competent they are in making a prediction for x. The overall prediction for $M(x)$ is obtained by linearly combining all the experts' predictions for x according to competency weights returned by the gate. In Sparse MoEs [13], this formulation is adapted to activate only a given number k of experts, selected as those that were assigned the top-k competency weights, for x, by the gate.

Model *OPM* The outcome-prediction neural-network model proposed in our approach, named *Mixture-of-Experts-OPM*, can be regarded as a tuple $\mathcal{N} = \langle \mathcal{N}_g, \mathcal{N}_1, \ldots, \mathcal{N}_m \rangle$ where $\mathcal{N}_E$ is the sub-net consisting of all its experts $\mathcal{N}_1, \ldots, \mathcal{N}_m$, and $\mathcal{N}_g$ is the gate sub-net. This model as a whole encodes a function $f : \mathbb{R}^d \rightarrow [0,1]$ defined as follows: $f(x) \triangleq \mathcal{N}_k(x)$ such that $k = \arg\max_{k \in \{1,\ldots,m\}} \mathcal{N}_g(x)[k]$ and $\mathcal{N}_g(x)[k]$ is the k-th component of the probability vector returned by the gate sub-net $\mathcal{N}_g$ when applied to x.

Thus, for any novel input instance x, a decision mechanism is applied to the output of the gate, which transforms it into an "argmax"-like weight vector where all the entries are zeroed but the one corresponding to the expert which received the the highest competency score (which is turned into 1); this makes the gate implement a hard-selection mechanism.

For the sake of interpretability, the following design choices are taken: *(i)* each expert $\mathscr{N}_i$, for $i \in [1..m]$ is implemented as one-layer feed-forward nets with linear activation functions, followed by a standard sigmoid transformation: *(ii)* the gate $\mathscr{N}_g$ is implemented as a one-layer feed-forward network with linear activation functions followed by a softmax normalization layer.

Training algorithm: ***MoE-OPM Discovery*** The proposed training algorithm, named *MoE-OPM Discovery*, takes two auxiliary arguments as input: the desired number m of expert sub-nets and the maximal number $kTop \in \mathbb{N} \cup \{\text{ALL}\}$ of input features per gate/expert sub-net (where ALL means that no actual upper bound is fixed for the latter number).

The algorithm performs four main steps: **(1)** A *Mixture-of-Experts-OPM* instance M is created, according to the chosen number m of experts, and initialized randomly. **(2)** M is trained end-to-end using an batch-based SGD-like optimization procedure (using different learning rates for the gate and the experts and a variant of the loss function proposed in [12], favouring expert specialization and competency weights' skeweness). **(3)** M is optimized again with the same procedure but keeping the expert sub-nets frozen, to fine-tune the gate one only. **(4)** Feature-wise parameter pruning is performed on both the gate and experts to make all these LR-like sub-nets base their predictions on $kTop$ data features at most.

The loss function utilized in the training algorithm combines an accuracy term alike the one proposed in [12] (favoring expert specialization) with a regularization term summing up the absolute values of all the model parameters. The influence of this regularization term can be controlled via weighting factor λ_R.

The last step of the algorithm consists in applying an ad hoc, magnitude-based, structured parameter pruning procedure to both the gate $\mathscr{N}_g$ and all experts $\mathscr{N}_1, \ldots, \mathscr{N}_m$. In this procedure, each parameter block gathers the weights of the connections reachable from a distinct input neuron, and all the parameters that do not belong to any of $kTop$ blocks are eventually zeroed. This corresponds to making all the sub-nets $\mathscr{N}_g$, $\mathscr{N}_1, \ldots, \mathscr{N}_m$ to only rely on $kTop$ input features.

3. Experiments

Algorithm *MoE-OPM Discovery*was tested against dataset extracted from the benchmark log *BPIC 2011* and *Sepsis*, obtained by making each prefix trace undergo the aggregation encoding after extending them all with timestamp-derived temporal features (e.g., weekday, hour, etc.). Each dataset was partitioned into training, validation, and test sets by using the same 80%-20%-20% temporal split as in [10, 2, 9]. The accuracy of each discovered model was evaluated by computing the AUC score for all test prefixes containing at least two events, as done in [10, 2, 9].

Algorithm *MoE-OPM Discovery* was using 100 epochs in both the end-to-end training and in the gate fine-tuning steps, without performing any post-pruning training (for the sake of efficiency). The number m of experts was fixed to 6 empirically (after trying several values in $[2, \ldots, 16]$), since this choice ensured a good accuracy-vs-simplicity trade-off. Different

Table 1
AUC scores obtained by: *MoE-OPM Discovery* (run with $m = 6$ and several values of $kTop$), the baseline method *1-LR* and two competitors. For each dataset, the best score is shown in **Bold and underlined**; *MoE-OPM Discovery*'s scores are shown in **Bold** when it outperforms the competitors.

Dataset	*MoE-OPM Discovery*					Competitors		
	kTop					*1-LR*	*FOX [10]*	*GLRM [9]*
	2	4	6	8	ALL			
bpic2011_1	0.97	0.95	0.96	**0.98**	0.88	0.94	0.97	0.92
bpic2011_2	0.85	0.84	0.86	**0.97**	0.87	0.94	0.92	**0.97**
bpic2011_3	0.95	**0.98**	0.96	**0.98**	0.91	0.97	**0.98**	**0.98**
bpic2011_4	0.69	0.81	0.80	0.81	0.80	0.68	**0.89**	0.81
sepsis_1	0.49	0.55	0.56	**0.58**	0.49	0.47	**0.58**	0.47
sepsis_2	0.56	0.56	**0.75**	0.73	0.72	0.74	0.73	0.73
sepsis_3	0.56	0.61	**0.72**	**0.72**	0.69	0.70	0.68	0.65

configurations were tested instead for hyperparameter λ_R (from 0.1 to 0.6) and $kTop$ (from 2 to 8). Details on the setting of other parameters (e.g., batch sizes, learning rates can be found in [1].

As terms of comparison, we considered state-of-the-art outcome-prediction methods *FOX* [10] and *GLRM* [9], and a baseline method, denoted as *1-LR*, that discovers a single LR model —the latter was simulated by running Algorithm *MoE-OPM Discovery* with $m = 1$ and $kTop = ALL$.

Prediction accuracy results Table 1 reports the AUC scores obtained by the 6-expert *MoE-OPM* models. Notably, *MoE-OPM Discovery* often outperforms the baseline *1-LR* in different $kTop \neq$ ALL configurations over some datasets, namely bpic2011_1, bpic2011_4, sepsis_1 . For the remaining datasets, there is always at least one $kTop$ configuration where *MoE-OPM Discovery* performs better than the baseline, when combined with feature selection. In particular, on average, *MoE-OPM Discovery* achieves an AUC improvement of more than 20% over *1-LR*, with peaks reaching beyond 80%. This confirms that training multiple local LR outcome predictors usually improves the performance of training a single LR model on all the data features (as done by *1-LR*). In addition, *MoE-OPM Discovery* always surpasses state-of-the-art methods FOX and GLRM on all the datasets but bpic2011_2 and bpic2011_3, where some of them perform as well as *MoE-OPM Discovery*.

Generally, *MoE-OPM Discovery* seems to perform worse when using very few features than when trained with a slightly larger feature set (namely, $kTop = 6, 8$). However, on dataset bpic2011_1, *MoE-OPM Discovery* manages to achieve outstanding AUC scores even when using just two (resp. four) features. The compelling AUC results obtained by the proposed approach using less than 9 input features per sub-model provides some empirical evidence of its ability to support the discovery of more compact outcome predictors and easier-to-interpret prediction explanations compared to the state-of-the-art method FOX, as discussed below.

Model/explanation complexity Generally, the lower the description complexity of a prediction model, the easier to interpret it and its predictions. In the cases of *MoE-OPM Discovery* and baseline *1-LR* this complexity can be computed by counting the non-zero parameters appearing in the respective LR (sub-)models, while the complexity of FOX model is the number of conditions appearing in its fuzzy rules. When applied to the pre-filtered datasets bpic2011_1, bpic2011_1,

Expert	0	1	2	3	4	5
Activity_CRP	0,00	0,00	0,00	-0,44	0,00	0,35
Activity_IV Liquid	0,85	0,00	-0,47	0,00	0,00	0,00
Activity_Release B	-0,32	-0,40	0,00	0,00	0,00	0,00
DiagnosticArtAstrup	0,00	0,40	0,00	0,00	0,00	0,00
DiagnosticSputum	0,00	0,00	0,00	-0,36	0,00	0,00
Hypotensie	0,00	0,00	0,00	0,00	-0,38	0,00
Oligurie	0,00	0,00	0,00	0,00	0,00	-0,37
SIRSCritHeartRate	0,00	0,00	0,00	0,00	-0,44	0,00
SIRSCritTemperature	0,00	0,00	0,00	0,00	-0,41	0,55
mean_hour	-0,33	0,00	0,00	0,00	0,00	0,00
mean_timesincelastevent	0,00	0,00	0,00	0,00	-0,36	0,00
org:group_A	0,00	0,00	-0,49	0,00	0,00	0,00
org:group_G	0,00	-0,34	0,00	0,00	0,00	0,00
org:group_H	0,00	0,00	0,00	0,52	0,00	0,00
org:group_O	0,00	0,00	0,00	0,00	0,00	0,36
org:group_T	0,00	0,00	-0,46	0,00	0,00	0,00
org:group_V	0,00	0,00	-0,48	0,00	0,00	0,00
org:group_W	0,00	0,34	0,00	0,00	0,00	0,00
org:group_other	-0,34	0,00	0,00	0,00	0,00	0,00
std_timesincemidnight	0,00	0,00	0,00	-0,33	0,00	0,00

Figure 1: Example *MoE-OPM* discovered by *MoE-OPM Discovery* (with $kTop = 4$) from dataset `sepsis_3`: parameter weights of the six LR experts.

..., `bpic2011_4`, `sepsis_1`, ..., `sepsis_3` (containing 4, 7, 6, 2, 5, 4 and 6 data features, respectively), FOX finds a model consisting of 81, 2187, 729, 9, 243, 81 and 729 fuzzy rules [10], respectively. Thus, the complexity of these FOX models range from 18 to 15309.

Qualitative results: an example of discovered *MoE-OPM* Figure 1 shows the input features and associated weights that are employed by the six LR experts discovered when running algorithm *MoE-OPM Discovery* with $kTop = 4$ on dataset `sepsis_3`, for which the outcome-prediction task is meant to estimate the probability that an in-treatment patient will leave the hospital with the prevalent release type (i.e., 'Release A'). Only 20 of the 86 data features are used by the experts in total, but the specific feature subset of the experts differ appreciably from one another. In a sense, this means that the experts learned different input-output mappings (capturing different contex-dependent process-outcome use cases).

For instance, in predicting class 1, Expert 0 attributes a positive influence to 'Activity_IV Liquid' and negative influence to 'Activity_Release B' (a specific discharge type), 'mean_hour', and 'org:group_other' (a hospital group). Expert 1 attributes instead a positive influence to 'DiagnosticArtAstrup' (arterial blood gas measurement) and 'org:group_W' and negative influence to both 'Activity_Release B' and 'org:group_G'. Analogous interpretations can be extracted from the remaining expert models, which also focus on specific activities and hospital groups.

4. Discussion and Conclusion

The experimental results presented above confirm that the models discovered by *MoE-OPM Discovery* exhibit a compelling trade-off between the accuracy and explainability of the their outcome predictions. This descends from both the modularity and conditional-computation nature of *OPM* models (where only one specific expert is chosen to make a pre-

diction), and from the possibility to control the complexity of both the model and of their explanations (via hyperparameters m and $kTop$). In particular, by focus on a small number of feature importance scores, the used is allowed to easily inspect and assess the internal decision logic of the model and get simple, faithful explanations for its predictions.

Interesting directions of future work are: (i) converting LR-like sub-models returned by *MoE-OPM Discovery* into logic rules, (ii) tuning hyper-parameters $kTop$ and m automatically; (iii) evaluating the relevance of *OPM*s' explanations through user studies.

References

[1] F. Folino, L. Pontieri, P. Sabatino, Sparse mixtures of shallow linear experts for interpretable and fast outcome prediction, in: Intl. Conf. on Process Mining Workshops, Revised Selected Papers, 2024, pp. 141—-152.

[2] I. Teinemaa, M. Dumas, M. L. Rosa, F. M. Maggi, Outcome-oriented predictive process monitoring: Review and benchmark, ACM Trans. on Knowledge Discovery from Data 13 (2019) 1–57.

[3] E. Rama-Maneiro, J. Vidal, M. Lama, Deep learning for predictive business process monitoring: Review and benchmark, IEEE Trans. on Services Computing (2021).

[4] R. Galanti, et al., Explainable predictive process monitoring, in: Proc. of 2nd Intl. Conf. on Process Mining (ICPM'20), 2020, pp. 1–8.

[5] W. Rizzi, C. D. Francescomarino, F. M. Maggi, Explainability in predictive process monitoring: When understanding helps improving, in: Proc. of 18th Intl. Conf. on Business Process Management (BPM'20), 2020, pp. 41—-158.

[6] B. Wickramanayake, et al., Building interpretable models for business process prediction using shared and specialised attention mechanisms, Knowledge-Based Systems 248 (2022) 108773.

[7] M. Stierle, S. Weinzierl, M. Harl, M. Matzner, A technique for determining relevance scores of process activities using graph-based neural networks, Decision Support Systems 144 (2021) 113511.

[8] D. Slack, A. Hilgard, S. Singh, H. Lakkaraju, Reliable post hoc explanations: Modeling uncertainty in explainability, Advances in Neural Information Processing Systems 34 (2021) 9391–9404.

[9] A. Stevens, J. D. Smedt, Explainability in process outcome prediction: Guidelines to obtain interpretable and faithful models, arXiv:2203.16073, 2023.

[10] V. Pasquadibisceglie, G. Castellano, A. Appice, D. Malerba, Fox: a neuro-fuzzy model for process outcome prediction and explanation, in: Proc. of 3rd Intl. Conf. on Process Mining (ICPM'21), 2021, pp. 112–119.

[11] A. A. Ismail, S. Ö. Arik, J. Yoon, A. Taly, S. Feizi, T. Pfister, Interpretable mixture of experts for structured data, arXiv preprint arXiv:2206.02107 (2022).

[12] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, G. E. Hinton, Adaptive mixtures of local experts, Neural Computation 3 (1991) 79–87.

[13] N. Shazeer, et al., Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, in: Proc. Intl. Conf. on Learning Representations (ICLR), 2017, pp. 1–17.