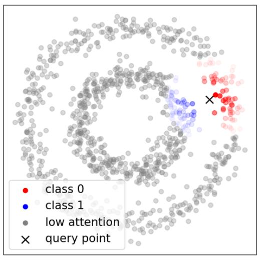
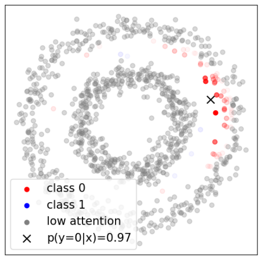
In-Context Learning of Soft Nearest Neighbor Classifiers for Intelligible Tabular Machine Learning

Mykhailo Koshil[1] **Matthias Feurer**[2,3] **Katharina Eggensperger**[1]
[1]University of Tübingen
[2]Department of Statistics, LMU Munich
[3]Munich Center for Machine Learning
first.last@uni-tuebingen.de, first.last@stat.uni-muenchen.de

Abstract

With in-context learning foundation models like TabPFN excelling on small supervised tabular learning tasks, it has been argued that "boosted trees are not the best default choice when working with data in tables".[1] However, such foundation models are inherently black-box models that do not provide interpretable predictions. We introduce a novel learning task to train ICL models to act as a nearest neighbor algorithm, which enables intelligible inference and does not decrease performance empirically.





(a) L_2 based similarity (b) SoftKNN-ICL similarity

Figure 1: Our in-context learning model makes predictions by weighting labels of similar data points (alpha value encodes weight). In contrast to L2-based nearest neighbor methods (left), our method learns the similarity function via in-context learning (right).

1 Introduction

In-context learning (ICL) yields state-of-the-art models for small supervised tabular learning tasks, exemplified by TabPFN (Hollmann et al., 2023, 2025). TabPFN is trained to solve supervised tabular learning tasks via in-context learning directly, meaning that at the inference time, the model is effectively fitted to the task without any weight updates. While such a model shows impressive performance, its inference mechanism is not interpretable, and users have to rely on model-agnostic explainability methods (Rundel et al., 2024). This is in contrast to recent requirements for more transparent, interpretable, and intelligible models (Rudin, 2019). [2]

K-Nearest neighbor (KNN) algorithms, a complementary research direction, recently reappeared in tabular state-of-the-art methods, such as ModernNCA (Ye et al., 2025) and TabR (Gorishniy et al., 2024). KNN-based methods make predictions based on the similarity between a query and training samples, thus offering transparent, example-driven inference. However, the performance of KNN is highly dependent on a similarity function and the choice of hyperparameter k, which are both dataset-specific, rendering this approach inappropriate for foundation models (FMs) working across many different datasets. The generalization to the soft-nearest neighbor method (Goldberger et al., 2004) bases its predictions on the weighted sum of the labels of all training samples in the dataset, yielding accurate predictions while still being human-interpretable. The ModernNCA extension (Ye et al., 2025) demonstrates that learning the similarity function via a neural network can further boost the performance.

In our work, we aim to obtain intelligible, state-of-the-art, off-the-shelf models and study *"How can we leverage nearest neighbor methods to make ICL more intelligible?"*

More precisely, we propose a novel training task for tabular ICL models, inspired by continuous nearest neighbor methods (Ye et al., 2025) and RAG (Gorishniy et al., 2024) (see Figure 1). Our contributions are the following:

1. We introduce a novel training task for ICL, yielding an intelligible extension for any tabular ICL model. We dub our method SoftKNN-ICL.

[1]https://bsky.app/profile/sammuller.bsky.social/post/3lfaql7hyhk2j

[2]See Vaughan and Wallach (2021) for a discussion of the term "intelligibility".

Proceedings of the 4th Table Representation Learning Workshop, pages 182–191
July 31, 2025 ©2025 Association for Computational Linguistics

2. We qualitatively and quantitatively evaluate our method on standard tasks and demonstrate it achieves competitive performance while being intelligible.

The following section discusses related literature on ICL for tabular data, nearest neighbor methods in deep learning and intelligible deep learning methods. After introducing and evaluating our method in Section 3 and Section 4, we further discuss how our method relates to kernel learning in Section 5 and conclude with future work and limitations in Section 6.

2 Related Work

In-context learning for tabular data. One of the successful paradigms for training tabular deep learning (DL) is ICL, where a model is trained on many datasets to make predictions for a test (query) set conditioned on the train (support) set. Interestingly, the ICL regime in the model competes with usual, in-weight learning, and has a transient nature (Singh et al., 2023). Early works on ICL for tabular data were developed for specific tasks (Garnelo et al., 2018a,b). Later work demonstrated that training these models using purely synthetic data can achieve strong performance (Müller et al., 2022; Hollmann et al., 2023; den Breejen et al., 2024), but general pre-training on natural data is also possible (Ma et al., 2024). ICL models perform well and primarily differ in the data used for pre-training, e.g., real or synthetic data, and architectural design, e.g., cell-based attention (den Breejen and Yun, 2025), yielding continuous performance improvements over time (den Breejen et al., 2024; Hollmann et al., 2025; Qu et al., 2025a). We leverage this model class and propose a new training task.

Few works argue that ICL models such as TabPFN learn an efficient kernel (Nagler, 2023; McCarter, 2024), and we will discuss this connection in more detail in Section 5. In concurrent work to make TabPFN invariant to class order, Arbel et al. (2025) also noted this connection. Their resulting model leverages a technique similar to ours but further processes a combination of labels with a non-linear module, because the main emphasis of their work is performance rather than intelligibility.

Finally, a complementary research direction leverages the ICL capability of LLMs for tabular data, instead of training FMs on tabular data (Gardner et al., 2024). While they perform well for small datasets, they are computationally expensive, not robust to table manipulations, and inherently struggle with large tables (Fang et al., 2024).

Development of nearest neighbor algorithms (NNA) in deep learning. NNAs are used extensively in deep learning models and mostly build on Nearest Component Analysis (NCA, Goldberger et al., 2004), also known as soft-NN, to allow for back-propagation. In NCA, the label for an unseen test sample is predicted by taking a weighted average of all available training samples. The follow-up work Nonlinear NCA (NNCA) (Salakhutdinov and Hinton, 2007) extends NCA to operate on features extracted with a neural network. The work of Vinyals et al. (2016) uses an NNA for few-shot learning and a bi-LSTM to capture global context. Plötz and Roth (2018) generalized this to a differentiable KNN selection rule, outputting a set of neighbors, rather than their average. Wang and Sabuncu (2023) study explainability of soft-NN methods for image classification from the perspective of the kernel methods. Recently, Li et al. (2024) proved that a 1-NN can be learned in-context with a one-layer transformer. Our model continues this line of research and is the first to explicitly combine NNAs with ICL, by training an ICL embedder that captures global context and produces features for a soft-NN.

Applications of NNAs in deep learning. The use of NNAs can be broadly categorized into two groups: those where NNAs serve as the core model and those where they enhance the performance of a downstream model. Retrieval-Augmented Generation (RAG) is a prominent method that improves the performance of an LLM by enriching the context with relevant information from an external knowledge base (Lewis et al., 2020). NNAs are also employed for scaling prompt size in LLMs (Xu et al., 2023; Zhao et al., 2024), and context localization in tabular ICL models, helping to relax the support set size limitations (Koshil et al., 2024; Thomas et al., 2024; Nejjar et al., 2024; Xu et al., 2025). Examples of the models with NNA as a core algorithm include the extended version of NNCA, ModernNCA (Ye et al., 2025), and TabR (Gorishniy et al., 2024), which is inspired by RAG. Our method SoftKNN-ICL also falls within the category of models using NNA at its core.

Intelligibility in deep learning. A model's decisions can be made intelligible either by designing the model to be explainable from the outset (intrinsic interpretability) or by applying post hoc explanation methods after training, which often entail a computational overhead or require a separate dataset. Basic DL models like MLP, ResNet, or Transformer are not intrinsically interpretable and require post-hoc explanation methods (Molnar, 2025) like SHAP (Lundberg and Lee, 2017) or LIME (Ribeiro et al., 2016). However, by combining neural networks with explainable methods like GAM (Chang et al., 2022), it is possible to leverage the complex features of DL models while maintaining intrinsic explainability. A more exotic approach is to train a deep learning meta-model that predicts the optimal parameters of an explainable model (Müller et al., 2023; Mueller et al., 2024), which, however, are constrained in size. Our model also combines an ICL transformer with an NNA model, which is considered intelligible if the features of the sample are/or can be made interpretable, e.g., by dimensionality reduction (Molnar, 2025).

3 Methodology

We are interested in supervised tabular classification, which is the task to predict test labels $\mathbf{y}^q \in \{c \in \mathbb{N} : c \leq C\}^m$ given p features of m test samples $\mathbf{X}^q \in \mathbb{R}^{m \times p}$ and a training set $(\mathbf{X}^s, \mathbf{y}^s)$, where $\mathbf{X}^s \in \mathbb{R}^{n \times p}$ and $\mathbf{y}^s \in \{c \in \mathbb{N} : c \leq C\}^n$.

Here, we focus on ICL approaches, which means a pre-trained model f_θ is "fitted" on the data set during the inference without weight updates, in contrast to the classical in-weight learning approach. To disambiguate the terminology, when talking about inference, we refer to the test set as *query* and the training set as *support*.

We introduce a novel learning task that implements a nearest neighbor method. KNN is the most popular nearest neighbor method and operates by assigning each query point a label y_j based on the majority vote of its k closest neighbors in the support set. This can be written down using an indicator function $\mathbb{1}_{\mathcal{N}}(x) := \{1 \text{ if } x \in \mathcal{N},\ \text{else } 0\}$, and defining a neighborhood $\mathcal{N}_j := \mathcal{N}(\mathbf{X}^q[j], \mathbf{X}^s, k)$ as a function returning a set of nearest neighbors according to a similarity function, most commonly based on the Euclidean distance. Then, the predicted label is:

$$\hat{y}_j = \arg\max_{c \in C} \sum_{i=1}^{n} \frac{ohc(\mathbf{y}^s)[i]\mathbb{1}_{\mathcal{N}_j}(\mathbf{X}^s[i])}{k}$$

with $\mathrm{ohc}(\mathbf{y}^s) = \{0,1\}^{n \times C}$ being the one-hot-encoded labels of the support set.

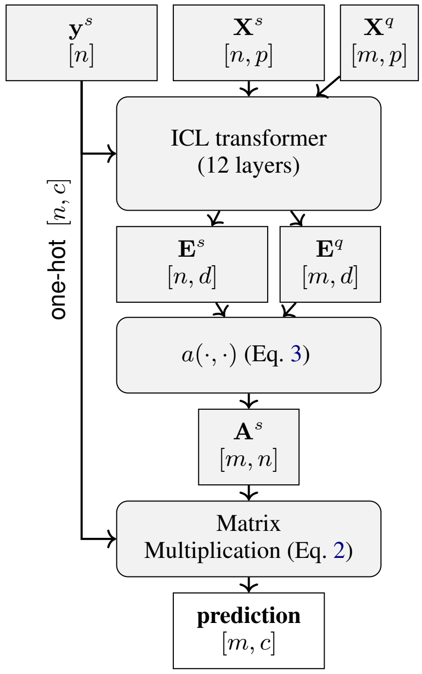


Figure 2: The architecture of SoftKNN-ICL. At the core of our approach is an ICL transformer that produces embeddings used to compute similarities between the query and support samples. The final prediction is obtained by taking a similarity-weighted average of the support labels.

However, we cannot directly leverage this as a learning task to fit a model, since the neighborhood function $\mathcal{N}(\cdot,\cdot,\cdot)$ is not differentiable. Instead, we propose to train the model using a continuous generalization of the KNN model (Goldberger et al., 2004) by allowing all data points to contribute to the prediction according to their similarity $a_j(\mathbf{X}^s[i]) = a(\mathbf{X}^q[j], \mathbf{X}^s[i]) := \mathrm{sim}(\mathbf{X}^q[j], \mathbf{X}^s[i])$:

$$\hat{\mathbf{y}}_j = \arg\max_{c \in C} \frac{\sum_{i=1}^{n} ohc(\mathbf{y}^s)[i] a_j(\mathbf{X}^s[i])}{\sum_{i=1}^{n} a_j(\mathbf{X}^s[i])}. \quad (1)$$

Now, the prediction is the weighted average of all labels in the support set, similar to Nadaraya-Watson kernel regression (Nadaraya, 1964; Watson, 1964), with the main difference that we do not explicitly condition the similarity function on the distance between inputs. We parametrize the similarity function by introducing an embedding function f_θ mapping the raw data to a latent space using information from the support and query sets: $a(f_\theta(\mathbf{X}^s, \mathbf{y}^s, \mathbf{X}^q)) \rightarrow \mathbf{A}^s, \mathbf{A}^s \in [0,1]^{m \times n}$. This allows learning a similarity function based on the given learning task, and we will explain later how to use a standard transformer architecture for this. Assuming similarity scores are normalized wrt.

support $|\mathbf{A}^s[i,\cdot]|_1 = 1$, prediction (1) can be written in matrix form:

$$\hat{\mathbf{y}}^q = \mathbf{A}^s \cdot \mathrm{ohc}(\mathbf{y}^s), \hat{\mathbf{y}}^q \in \{0,1\}^{m \times C}, \quad (2)$$

where labels can be obtained as $\hat{\mathbf{y}}_j = \arg\max_{c \in C} \hat{\mathbf{y}}^q[j,\cdot]$ This setup directly conceptually matches ICL, which operates on support and query sets. However, existing ICL models are not (yet) explicitly trained to make predictions by weighting support set labels.

We propose implementing the similarity function a by taking the transformed embeddings of the query and the support set and a merged $\mathbf{KV}^T$ matrix of a transformer layer $\mathbf{W} \in \mathbb{R}^{d \times d}$:

$$a(\mathbf{E}^q[j], \mathbf{E}^s) := \mathrm{softmax}((\mathbf{E}^q(\mathbf{W} \cdot \mathbf{E}^{sT}))[j,\cdot]), \quad (3)$$

with $f_\theta(\mathbf{X}^s, \mathbf{y}^s, \mathbf{X}^q) \rightarrow (\mathbf{E}^s, \mathbf{E}^q)$, $\mathbf{E}^s \in \mathbb{R}^{n \times d}$ and $\mathbf{E}^q \in \mathbb{R}^{m \times d}$ being the corresponding embeddings of $\mathbf{X}^s$ and $\mathbf{X}^q$ with dimensionality d. Thus, $\mathbf{a}^q := \mathbf{A}^s[q]$ is a corresponding row of the "attention matrix" representing attention values from query $\mathbf{x}^q$ to the support samples $\mathbf{X}^s$. The merged $\mathbf{KV}^T$ matrix follows work on learning KNN via ICL with linear transformers (Li et al., 2024). This means that we train our transformer model, for a given query point, *to attend to similar points in the support set* and to make predictions by weighting the labels of all points in the support set based on these similarity (attention) values. For training our model, we use the cross-entropy loss $L(\hat{\mathbf{y}}^q, \mathbf{y}^q) = \mathrm{CE}(\hat{\mathbf{y}}^q, \mathbf{y}^q)$. We refer to this model as SoftKNN-ICL and display its structure in Figure 2.

We also experimented with the alternative, potentially more straightforward, implementation which outputs the 1-d logit per token by setting $m = 1$ and $d = 1$ and taking a softmax over the sample dimension, $a(\mathbf{E}^s, \mathbf{E}^q) := \mathrm{softmax}(\mathbf{E}^s[\cdot, 1])$. However, this version results in inferior convergence and requires advanced pre-training schedules, so we do not consider it further.

Implementation and Hardware Details. Our implementation is based on the repository of den Breejen et al. (2024), and we will release our code upon acceptance.[3] Following other works in the field, e.g., Hollmann et al. (2023) and den Breejen et al. (2024), the model is trained using synthetic data only. Concretely, we use the TabForest prior as introduced by den Breejen et al. (2024), which is a mix of the original TabPFN prior (Hollmann et al., 2023) and the forest prior (den Breejen et al., 2024). In practice, we add information from the label in $\mathbf{X}^s$ as part of the input token, following the standard TabPFN methodology. Optimization is performed using Adam (Kingma and Ba, 2015) with learning rate of $4\mathrm{e}{-5}$. We employ cosine annealing (Loshchilov and Hutter, 2017) with linear warmup (10 epochs with 8192 datasets) for learning rate scheduling. SoftKNN-ICL is trained using three V100 GPUs on 24.6M synthetic datasets.

4 Experimental Evaluation

We divide the evaluation of our model into two parts. First, we perform a study using toy problems to analyze the decision boundaries of our model. Second, we compare our model against competitor models on standard benchmark datasets.

4.1 Decision Boundaries on Toy Problems

First, we want to study how our model behaves on simple toy problems. In Figure 3 we compare decision boundaries on 2-dimensional toy datasets of our SoftKNN-ICL to KNN (using $k = 3$) and the Nadaraya-Watson estimator (using RBF kernel with $\gamma = 15$) as the methodologically closest non-deep-learning methods. Furthermore, we compare against an SVM (using RBF kernel with $\gamma = 5, C = 3$) and TabForestPFN (den Breejen et al., 2024). Overall, SoftKNN-ICL yields competitive performance and reasonable decision boundaries. Compared to the nearest neighbor methods (second and third column), our method provides reasonable uncertainty estimates when moving away from seen datapoints (see "Moons" and "Circles") and is less prone to overfitting on noisy datasets (see "Noisy Moons" and "Noisy Circles"). Additionally, it performs comparably to the TabForestPFN model, which is desirable.

Furthermore, we study the neighborhood used to make predictions. In the last column of Figure 3, we visualize the values of $\mathbf{a}^q$ (see Equation (1)), i.e., the predicted similarity between the query point (black cross) and the data set. Overall, the neighborhood of SoftKNN-ICL can become very small, with the bias of selecting samples from the same class (see "Circles"). The most interesting finding is that the model dynamically adjusts the number of samples it considers for prediction: when the neighborhood is noisy (i.e., the nearest samples do

[3] https://github.com/FelixdenBreejen/TabForestPFN

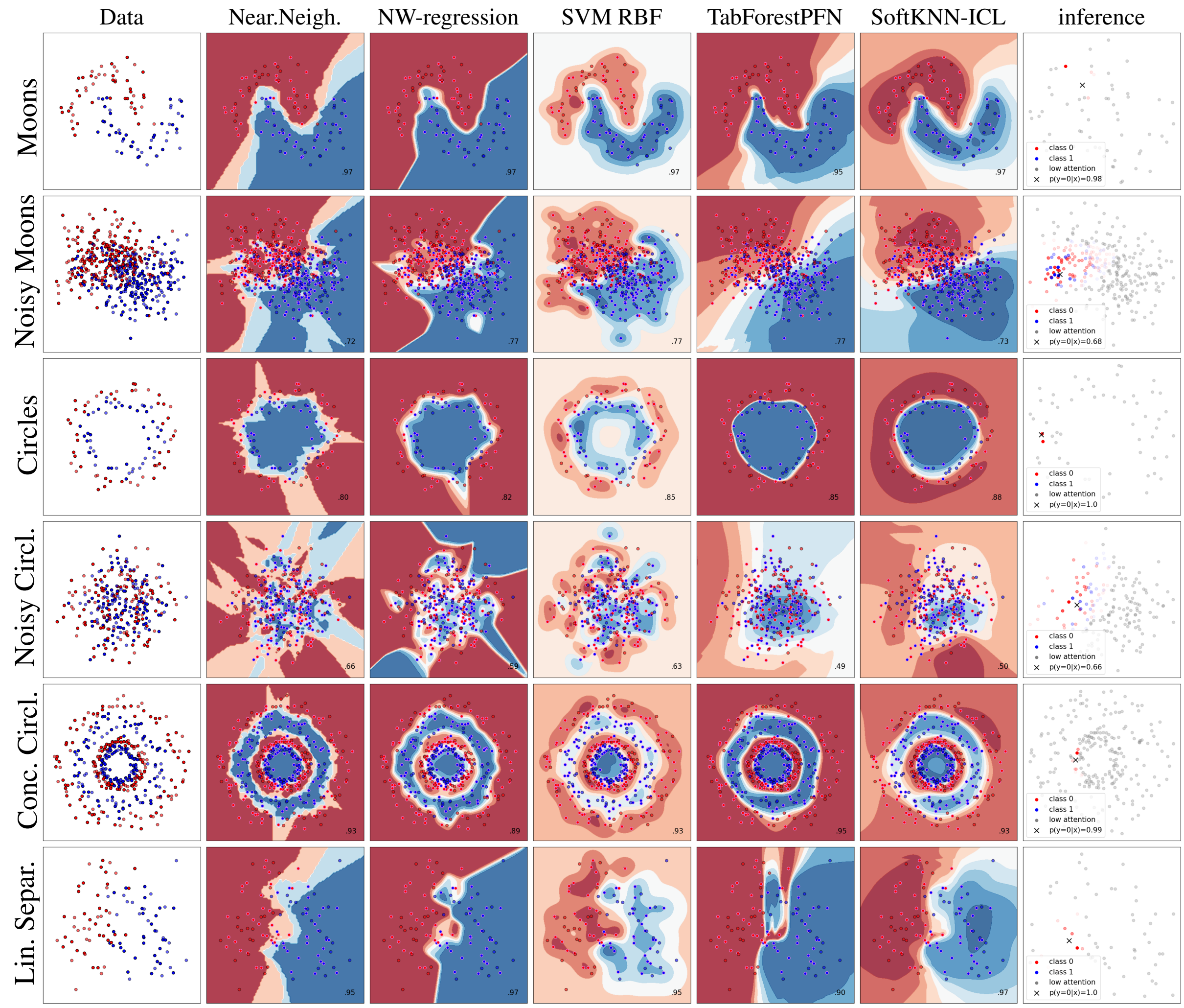


Figure 3: Decision boundary of SoftKNN-ICL and other methods on toy datasets.

not exhibit a dominant class), it aggregates information from more points, similar to decreasing γ in an RBF kernel. In contrast, when the nearest sample is strongly indicative, the model relies primarily on the labels of a few samples (compare "Moons" and "Circles" with "Noisy Moons" and "Noisy Circles").

4.2 Evaluation on Real-World Datasets

Next, we compare our method against baselines using standard benchmark tasks. Concretely, we use the same datasets as the TabPFN paper (Hollmann et al., 2023): these are 30 datasets from the OpenML benchmarking suites *CC-18* (Bischl et al., 2021), restricted to contain at most 2 000 data points. Inspired by the original evaluation protocol, which uses five randomized $50/50$ train/test splits, we conducted a two-fold cross-validation five times to reduce the variance of our results by guaranteeing that each datapoint is used for testing in each repetition while using training and test sets of the same size as in the original evaluation protocol. We provide OpenML task IDs in Table 2 in Appendix A to allow reproducing our results. We compare average AUC across all repetitions and datasets.

As baselines, we use the TabPFN model provided by Hollmann et al. (2023) and the TabForestPFN model provided by den Breejen et al. (2024), which is trained with the same TabForest prior (den Breejen et al., 2024) as our model SoftKNN-ICL. Additionally, we disable ensembling by input permutations for all PFN-style models.[4] To test the capabilities of the nearest neighbor algorithm, we also use a traditional KNN with $k = 1$ and $k = 5$ from scikit-learn (Pedregosa et al., 2011), where we preprocess the data as it is

[4]Enabling ensembling could further boost our performance, but this is not the goal of our study. Furthermore, ensembling would decrease the intelligibility of our proposed method.

Model Name	k	avg. AUC
Random Forest	n.a.	0.8712
TabForestPFN	n.a.	0.8816
TabPFN	n.a.	**0.8856**
KNN	1	0.7498
	5	**0.8272**
SoftKNN-ICL (ours)	1	0.7746
	5	0.8460
	10	0.8606
	all	**0.87975**

Table 1: Average AUC of all methods on 30 datasets using 5-repeated 2-fold cross-validation. We boldface the best method in each category.

in the original evaluation protocol (Hollmann et al., 2023).

We present average AUC values in Table 1. Notably, SoftKNN-ICL outperforms KNN with different values of K and matches the performance of TabPFN and the TabForestPFN trained on the same synthetic datasets. Furthermore, in Figure 4 we compare AUC values per dataset, showing that there are no outlier datasets on which SoftKNN-ICL performs substantially better or worse than the current PFN architecture. Lastly, Figure 5 reports the average ranks and statistical results following Demšar (2006), demonstrating that our SoftKNN-ICL does not perform statistically differently than TabForestPFN and TabPFN.

We also conducted an ablation on using only the top-k similar datapoints from the support set (as done by Wang and Sabuncu (2023)). While performance (not surprisingly) degrades, it is better than for KNN with the same number of neighbors, and using only a fixed number of neighbors could be valuable for tasks where it is essential to be able to study which samples contribute to the prediction.

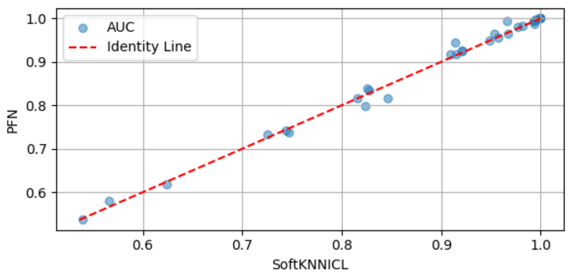


Figure 4: AUC values of SoftKNN-ICL vs. PFN. Each dot corresponds to one dataset.

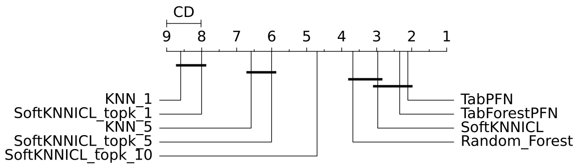


Figure 5: Average rank and critical distance diagram.

5 Connection with kernel machines and metric learning

Before turning to the conclusion and after having presented the technical details of SoftKNN-ICL, we would like to embed our method further into the existing literature. NCA inspired our method; however, our methodological framework allows us to connect our method and the fields of metric learning and kernel learning (Bellet et al., 2013), which we briefly highlight in the following. As shown in Equation 3, our model effectively performs kernel regression and can be framed as a deep kernel learning with an exponential kernel as the base kernel (see Equation (5) in Wilson et al. (2016)). While it is known that self-attention mechanisms can be interpreted through the lens of kernel methods (Tsai et al., 2019), this connection opens up promising directions for future research. These include exploring alternative base kernels for the final layer, or gaining insight into the mechanisms of ICL by revisiting the approach of Han et al. (2024). Their work developed a theoretical and empirical framework for studying this phenomenon and its connection to kernel methods in LLMs. Our settings are more constrained than in the original work (our model is sample-order invariant and can be made feature-order invariant using the attention mechanism proposed by den Breejen and Yun (2025)), which helps to mitigate some of the issues raised in reviews. Furthermore, the model can be reformulated as a metric learning approach by expressing the final layer (before normalization wrt support dimension) as $\mathbf{A}^{\text{unnorm}}[\mathbf{X}^q[j], \mathbf{X}^s[i]] = exp(-||\mathbf{W}(\mathbf{E}^q[j] - \mathbf{E}^s[i])||^2)$, following the formulation in (Weinberger and Tesauro, 2007). This makes the model to explicitly learn a metric between the support and query points $d((\mathbf{X}^s, \mathbf{y}^s), (\mathbf{X}^q)) = ||\mathbf{W}(\mathbf{E}^q[j] - \mathbf{E}^s[i])||$ in the embedding space parametrized by the embedder f_θ (ICL-transformer in our model) and $\mathbf{W}$. Connecting to a growing body of literature that seeks to relate kernel methods and neural networks (Belkin et al., 2018; Domingos, 2020;

Bell et al., 2023; Tarzanagh et al., 2023; Teo and Nguyen, 2024; Wilson, 2025; Arbel et al., 2025), our model could largely benefit from the synergy between both fields.

6 Conclusion and Future Work

We have demonstrated that a (soft) KNN learning task for ICL models leads to competitive performance compared to the standard learning task. The resulting SoftKNN-ICL is closely related to kernel and metric learning and can be used as a drop-in replacement for tasks requiring intelligibility. Additionally, by using SoftKNN-ICL, we overcome two limitations of traditional KNN methods: (1) the need to tune the number of neighbors, k, and the need to define a neighborhood (similarity) function manually. We hope this spurs research into interpretability methods targeted at instance-based learning methods, and that the in-context learning of a soft neighborhood is a valuable basis for distance learning, potentially even beyond tabular tasks. Furthermore, we deem future work along the following directions particularly interesting for tabular machine learning.

Detailed empirical evaluation. Most importantly, we plan to study how our method uses attention in noisy query sets and how different data-generating priors, used to train the ICL model, impact performance and behaviour.

Alternative architecture and learning tasks. Secondly, by extending our methodology of ICL using neighbor methods to, for example, using the NCA prediction function or training the model without the merged $\mathbf{KV}^T$ matrix, we hope to understand better how to train an ICL nearest neighbor method in the best manner. Other possible architecture improvements include the use of cell-based attention like in TabPFN v2 (Hollmann et al., 2025) and TabICL (Qu et al., 2025b), efficient embeddings similar e.g. TabICL, and localization methods (Thomas et al., 2024; Koshil et al., 2024) to mitigate the need of ensembling and improve scaling wrt. training set.

Making use of the distance function. Finally, while we only assessed the learned distance function to make predictions, it should also be possible to use it for exploratory data analysis and meta-learning. Additionally, it would be interesting to condition our method to consider as few neighbors as possible.

Limitations

Firstly, our method inherits the limitations of the ICL model class it resembles, i.e., limited context size and slow inference speed. Secondly, it is not as powerful as TabPFN (yet); however, we expect it to improve with longer training and hyperparameter tuning. Thirdly, our evaluation of intelligibility is limited to synthetic datasets; a thorough evaluation, potentially including a user study, remains future work. Lastly, although our model's inference mechanism is transparent by explicitly combining labels of existing data points, it remains unclear why these points are chosen due to the black-box nature of transformer models.

Acknowledgments

Katharina Eggensperger and Mykhailo Koshil acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645 and by the Baden-Württemberg Ministry of Science and the Federal Ministry of Education and Research (BMBF) as part of the Excellence Strategy of the German Federal and State Governments. The authors also thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Mykhailo Koshil. Last but not least, the authors thank Thomas Nagler for insightful discussions that led to the developments in this paper.

References

M. Arbel, D. Salinas, and F. Hutter. 2025. EquiTabPFN: a target-permutation equivariant prior fitted networks. *arXiv:2502.06684 [cs.LG].*

M. Belkin, S. Ma, and S. Mandal. 2018. To understand deep learning we need to understand kernel learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, volume 80, pages 541–549. Proceedings of Machine Learning Research.

B. Bell, M. Geyer, D. Glickenstein, A. Fernandez, and J. Moore. 2023. An exact kernel equivalence for finite classification models. In *Proceedings of 2nd Annual Workshop on Topology, Algebra, and Geometry in Machine Learning (TAG-ML)*, volume 221 of *Proceedings of Machine Learning Research*, pages 206–217. PMLR.

A. Bellet, A. Habrard, and M. Sebban. 2013. A survey on metric learning for feature vectors and structured data. *arXiv:1306.6709 [cs.LG].*

B. Bischl, G. Casalicchio, M. Feurer, F. Hutter, M. Lang, R. Mantovani, J. van Rijn, and J. Vanschoren. 2021. OpenML benchmarking suites. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Curran Associates.

C.-H. Chang, R. Caruana, and A. Goldenberg. 2022. NODE-GAM: Neural generalized additive model for interpretable deep learning. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*. Published online: iclr.cc.

J. Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

F. den Breejen, S. Bae, S. Cha, and S.-Y. Yun. 2024. Fine-tuned in-context learning transformers are excellent tabular data classifiers. https://openreview.net/forum?id=pE0UM18TQh. Unpublished manuscript, rejected at ICLR'24.

F. den Breejen and S. Yun. 2025. Attic: A new architecture for tabular in-context learning transformers. https://openreview.net/forum?id=DSl9sSuUhp. Unpublished manuscript, rejected from ICLR'25.

P. Domingos. 2020. Every model learned by gradient descent is approximately a kernel machine. *arXiv:2012.00152 [cs.LG]*.

X. Fang, W. Xu, F. Tan, Z. Hu, J. Zhang, Y. Qi, S. Sengamedu, and C. Faloutsos. 2024. Large language models (LLMs) on tabular data: Prediction, generation, and understanding - a survey. *Transactions on Machine Learning Research*.

J. Gardner, J. Perdomo, and L. Schmidt. 2024. Large scale transfer learning for tabular data via language modeling. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'24)*, pages 45155–45205. Curran Associates.

M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. Teh, D. Rezende, and S. Eslami. 2018a. Conditional neural processes. In *Proceedings of the 35th International Conference on Machine Learning (ICML'18)*, volume 80, pages 1704–1713. Proceedings of Machine Learning Research.

M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. Rezende, S. Ali Eslami, and Y. Teh. 2018b. Neural processes. *arXiv:1807.01622 [cs.LG]*.

J. Goldberger, G. Hinton, S. Roweis, and R. Salakhutdinov. 2004. Neighbourhood components analysis. In *Proceedings of the 18th International Conference on Advances in Neural Information Processing Systems (NeurIPS'04)*. MIT Press.

Y. Gorishniy, I. Rubachev, N. Kartashev, D. Shlenskii, A. Kotelnikov, and A. Babenko. 2024. TabR: tabular deep learning meets nearest neighbors. In *International Conference on Learning Representations (ICLR'24)*. Published online: iclr.cc.

C. Han, Z. Wang, H. Zhao, and H. Ji. 2024. Explaining emergent in-context learning as kernel regression. https://openreview.net/forum?id=v9Pguuamfp. Unpublished manuscript, rejected at ICLR'24.

N. Hollmann, S. Müller, K. Eggensperger, and F. Hutter. 2023. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR'23)*. Published online: iclr.cc.

N. Hollmann, S. Müller, L. Purucker, A. Krishnakumar, M. Körfer, S. Hoo, R. Schirrmeister, and F. Hutter. 2025. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326.

D. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR'15)*. Published online: iclr.cc.

M. Koshil, T. Nagler, M. Feurer, and K. Eggensperger. 2024. Towards localization via data embedding for tabPFN. In *NeurIPS 2024 Third Table Representation Learning Workshop*.

P. Lewis, E. Perez, A. Piktus, F. Petroni, C. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and K. Douwe. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems (NeurIPS'20)*, pages 9459–9474. Curran Associates.

Z. Li, Y. Cao, C. Gao, Y. He, H. Liu, J. Klusowski, J. Fan, and M. Wang. 2024. One-layer transformer provably learns one-nearest neighbor in context. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'24)*, pages 82166–82204. Curran Associates.

I. Loshchilov and F. Hutter. 2017. SGDR: Stochastic gradient descent with warm restarts. In *Proceedings of the International Conference on Learning Representations (ICLR'17)*. Published online: iclr.cc.

S. Lundberg and S.-I. Lee. 2017. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'17)*. Curran Associates.

J. Ma, V. Thomas, R. Hosseinzadeh, H. Kamkari, A. Labach, J. Cresswell, K. Golestan, G. Yu, M. Volkovs, and A. Caterini. 2024. TabDPT: Scaling tabular foundation models. *arXiv:2410.18164 [cs.LG]*.

Calvin McCarter. 2024. What exactly has tabPFN learned to do? In *The Third Blogpost Track at ICLR 2024*.

C. Molnar. 2025. *Interpretable Machine Learning*, 3rd edition. Self-published.

A. Mueller, J. Siems, H. Nori, D. Salinas, A. Zela, R. Caruana, and F. Hutter. 2024. GAMformer: In-context learning for generalized additive models. *arXiv:2410.04560 [cs.LG]*.

Andreas Müller, Carlo Curino, and Raghu Ramakrishnan. 2023. Mothernet: A foundational hypernetwork for tabular classification. *arXiv:2312.08598 [cs.LG]*.

S. Müller, N. Hollmann, S. Arango, J. Grabocka, and F. Hutter. 2022. Transformers can do Bayesian inference. In *Proceedings of the International Conference on Learning Representations (ICLR'22)*. Published online: iclr.cc.

E. Nadaraya. 1964. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

T. Nagler. 2023. Statistical foundations of prior-data fitted networks. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, volume 202 of *Proceedings of Machine Learning Research*, pages 25660–25676. PMLR.

I. Nejjar, F. Ahmed, and O. Fink. 2024. IM-context: In-context learning for imbalanced regression tasks. *Transactions on Machine Learning Research*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

T. Plötz and S. Roth. 2018. Neural nearest neighbors networks. In *Proceedings of the 31st International Conference on Advances in Neural Information Processing Systems (NeurIPS'18)*. Curran Associates.

J. Qu, D. Holzmüller, G. Varoquaux, and M. Le Morvan. 2025a. Tabicl: A tabular foundation model for in-context learning on large data. *arXiv:2502:05564 [cs.LG]*.

J. Qu, D. Holzmüller, G. Varoquaux, and M. Le Morvan. 2025b. Tabicl: A tabular foundation model for in-context learning on large data. *Preprint*, arXiv:2502.05564.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1135–1144. Association for Computing Machinery.

C. Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, pages 206—215.

D. Rundel, J. Kobialka, C. von Crailsheim, M. Feurer, T. Nagler, and D. Rügamer. 2024. Interpretable machine learning for TabPFN. In *Explainable Artificial Intelligence*, volume 2154, pages 465–476.

R. Salakhutdinov and G. Hinton. 2007. Learning a nonlinear embedding by preserving class neighbourhood structure. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS'07)*, pages 412–419. Proceedings of Machine Learning Research.

A. Singh, S. Chan, T. Moskovitz, E. Grant, A. Saxe, and F. Hill. 2023. The transient nature of emergent in-context learning in transformers. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'23)*, pages 27801–27819. Curran Associates.

D. Tarzanagh, Y. Li, C. Thrampoulidis, and S. Oymak. 2023. Transformers as support vector machines. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*.

R. Teo and T. Nguyen. 2024. Unveiling the hidden structure of self-attention via kernel principal component analysis. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'24)*. Curran Associates.

V. Thomas, J. Ma, R. Hosseinzadeh, K. Golestan, G. Yu, M. Volkovs, and A. Caterini. 2024. Retrieval & fine-tuning for in-context tabular models. In *Proceedings of the 37th International Conference on Advances in Neural Information Processing Systems (NeurIPS'24)*, pages 108439–108467. Curran Associates.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer‘s attention via the lens of kernel. 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 4344–4353, Hong Kong, China. Association for Computational Linguistics.

J. Vanschoren, J. van Rijn, B. Bischl, and L. Torgo. 2014. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60.

J. Vaughan and H. Wallach. 2021. *A Human-Centered Agenda for Intelligible Machine Learning*, chapter 1. MIT Press.

O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. 2016. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Advances in Neural Information Processing Systems (NeurIPS'16)*. Curran Associates.

A. Wang and M. Sabuncu. 2023. A flexible Nadaraya-Watson head can offer explainable and calibrated classification. *Transactions on Machine Learning Research*.

G. Watson. 1964. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A (1961-2002)*, 26(4):359–372.

K. Weinberger and G. Tesauro. 2007. Metric learning for kernel regression. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS'07)*, pages 612–619. Proceedings of Machine Learning Research.

A. Wilson. 2025. Deep learning is not so mysterious or different. *arXiv:2503:02113 [cs.LG]*.

A. Wilson, Z. Hu, R. Salakhutdinov, and E. Xing. 2016. Deep kernel learning. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics (AISTATS'16)*, volume 51, pages 370–378. Proceedings of Machine Learning Research.

B. Xu, Q. Wang, Z. Mao, Y. Lyu, Q. She, and Y. Zhang. 2023. kNN prompting: Beyond-context learning with calibration-free nearest neighbor inference. In *International Conference on Learning Representations (ICLR'23)*. Published online: `iclr.cc`.

D. Xu, R. Asadi F Cirit, Y. Sun, and W. Wang. 2025. Mixture of in-context prompters for tabular pfns. In *International Conference on Learning Representations (ICLR'25)*. Published online: `iclr.cc`.

H. Ye, H.-Hong Yin, D. Zhan, and W. Chao. 2025. Revisiting nearest neighbor for tabular data: A deep tabular baseline two decades later. In *International Conference on Learning Representations (ICLR'25)*. Published online: `iclr.cc`.

W. Zhao, Y. Liu, Y. Wan, Y. Wang, Q. Wu, Z. Deng, J. Du, S. Loi, Y. Xu, and P. Yu. 2024. kNN-ICL: Compositional task-oriented parsing generalization with nearest neighbor in-context learning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 326–337. Association for Computational Linguistics.

A Dataset and Task IDs

Data ID	Dataset Name	Task ID
11	balance-scale	361412
14	mfeat-fourier	361414
15	breast-w	361415
16	mfeat-karhunen	361416
18	mfeat-morphological	361417
22	mfeat-zernike	361419
23	cmc	361420
29	credit-approval	363512
31	credit-g	233149
37	diabetes	361424
50	tic-tac-toe	363513
54	vehicle	361426
188	eucalyptus	363511
458	analcatdata_authorship	361437
469	analcatdata_dmft	363514
1049	pc4	363515
1050	pc3	363516
1063	kc2	361440
1068	pc1	363517
1462	banknote-authentication	361462
1464	blood-transfusion-...	361463
1480	ilpd	363518
1494	qsar-biodeg	361448
1510	wdbc	361442
6332	cylinder-bands	363519
23381	dresses-sales	363520
40966	MiceProtein	363521
40975	car	363522
40982	steel-plates-fault	363523
40994	climate-model-...	363524

Table 2: OpenML (Vanschoren et al., 2014) dataset and task IDs used for the evaluation.