RAM: Retrieval Augmented Modelling for Tabular In-Context Few-Shot Domain Adaptation

Oleh Kostromin, Felix Kossak, Michael Zwick *

Software Competence Center Hagenberg GmbH
Softwarepark 32a, 4232 Hagenberg, Austria

Abstract. Transformer architectures have shown great success in natural language processing, sparking interest in their applications on tabular data. However, the potential of using transformer-like architectures for in-context domain adaptation in tabular settings remains underexplored. We introduce Retrieval-Augmented Modelling (RAM), a compact attention-based architecture specifically designed for this task. RAM utilises a Domain-Aligned Memory training strategy, which ensures that it always processes the data from the same domain at each training step, allowing the model to focus on domain-specific patterns. Evaluated on synthetic data simulating domain shifts, RAM outperforms traditional machine learning models, effectively adapting to unseen domains.

1 Introduction

Transformers have achieved remarkable success in natural language processing tasks. They excel not only in general reasoning, but also in extracting information from external memory and performing in-context learning. This success has motivated attempts to adapt transformer-like models for tabular data. However, the suitability of transformer-like architectures for tabular in-context domain adaptation remains underexplored. Domain adaptation is crucial in scenarios where models need to generalise to new domains with limited labelled data - a common situation in real-world applications. Existing models either concentrate on within-domain improvements, or aim for universal generalisation across all tasks, potentially overshooting the specific objectives of domain adaptation and adding unnecessary complexity for practical use in resource-constrained environments.

To address this gap, we propose RAM, a compact attention-based architecture designed specifically for in-context domain adaptation on tabular data. RAM aims to balance the performance and size, making it suitable for CPU inference and deployment on mobile or edge devices. In addition to the architecture, we propose a Domain-Aligned Memory (DAM) training procedure. This training strategy ensures that the model's inputs and memory consist of data

*The research reported in this paper has been partly funded by BMK, BMAW, and the State of Upper Austria in the frame of the SCCH competence center INTEGRATE (FFG grant no. 892418) part of the FFG COMET Competence Centers for Excellent Technologies Programme.

from the same domain at each step, allowing the model to rely more effectively on the memory and focus on identifying domain-specific patterns in-context.

We evaluate RAM, along with the DAM training procedure, on a synthetic dataset specifically constructed to simulate domain shifts. We compare the model's performance to traditional machine learning approaches such as nearest neighbours, linear regression and tree-based methods. Our results demonstrate that models lacking in-context domain adaptation capabilities struggle to maintain performance under domain shifts and are outperformed by RAM. These findings highlight RAM's ability to effectively adapt to new domains, offering an advantage over models not equipped for in-context adaptation.

2 Related Work

2.1 In-Context Learning and Retrieval Augmented Generation

In-context learning (ICL) is the ability of the model to generalise to unseen tasks without being restrained by directly learning from several (few) examples provided directly as input during inference [1]. This technique is widely used in natural language processing by including a description of the task (instruction) and several solved cases in the prompt. Compared to supervised learning, ICL does not require parameter updates, which can assist with broadening the application area by using the same model for different downstream tasks.

The in-context learning mechanism has also been successfully applied in the tabular domain. As demonstrated by Holmann et al. [2], a single model pre-trained on a large volume of synthetically generated data can solve small classification tasks (up to 1000 samples and 100 features) while outperforming traditional models like Gradient-Boosted Decision Trees (GBDT) and achieving performance comparable to state-of-the-art AutoML systems but at a fraction of the time.

Retrieval-Augmented Generation (RAG) is a technique that allows to provide additional (external) information to the model during inference [3]. In the natural language domain, RAG has been widely used for open-domain question answering and other knowledge-intensive tasks that require accessing external information to generate accurate responses.

In the tabular domain, RAG was mainly used for allowing the models to refine the predictions by directly accessing the whole training set. Hopular [4], NPT [5] and SAINT [6] use stacked blocks of inter-sample and inter-feature interaction layers which allows to perform multiple memory lookups in a single inference pass. TabR [7] offers a simplified approach by using a single retriever module at the beginning. In addition, contrary to other approaches, TabR retrieval mechanism is based on Euclidean distance instead of the cosine similarity. These tabular RAG architectures are able to achieve state of the art performance among deep learning methods and compete with gradient-boosted decision trees.

Although RAG and ICL are two distinct concepts, we consider them to be highly interrelated as both enable the models to adapt to new information without additional training.

2.2 Compact Transformers

Reducing the size and inference latency of the transformer models while preserving their reasoning capabilities is an important research focus for applications with limited resources, like on-device deployments. Common strategies for model compression usually include quantization, where the weights are converted to lower precision representation while the architecture remains unchanged, and the usage of layers and blocks specifically designed for the reduction of the number of weights. Examples of such layers are Multi-Query Attention (MQA) [8] and Grouped-Query Attention (GQA) [9], variations of a regular self-attention mechanism that reuse the key-value projection matrices across all or several attention heads respectively, accelerating the computation speed-up by several times, albeit with some performance degradation. Another approach for constructing a more compact transformer architecture was proposed by Liu et al. [10], who suggested using narrower layers and sharing the weights across multiple blocks to build deeper models without increasing the weight count.

In the context of tabular transformers, the problem of model size and latency is equally important, given that transformer-based models often have to compete with traditional machine learning methods that offer similar predictive power while being more compact.

3 Retrieval Augmented Modeling

In this section, we introduce Retrieval-Augmented Modeling (RAM), our proposed architecture designed specifically for in-context domain adaptation on tabular data. RAM consists of four main components (shown on Figure 1): an embedding module, a retrieval module, a feature-transformer module and a task-specific head.

Embedding module maps features into a common embedding space. Following the input format of Hopular, this block operates on flattened vectors per sample, where numerical features are unit-scaled, and categorical features are one-hot encoded. The first layer maps the input features to an intermediate representation that is four times the size of the embedding dimension. The outputs of the first layer are activated using GELU. The second layer projects the intermediate representation into the target space and is shared across all features. Both layers are followed by a layer normalisation block. Additionally, type and positional encodings are optimally added to the embeddings.

Retriever performs the lookup in the external memory. Lookup is performed by a modified attention mechanism that, in addition to reusing value projections across all attention heads (as in multi-query attention), also uses a single shared projection matrix per head for both keys and queries. Following the idea of TabR, we use only a single retriever block immediately after the embedding block.

Feature transformer iteratively refines the representations obtained from the retriever module. Our implementation is inspired by MobileLLM [10] and includes several stacked transformer blocks with grouped-query attention. We

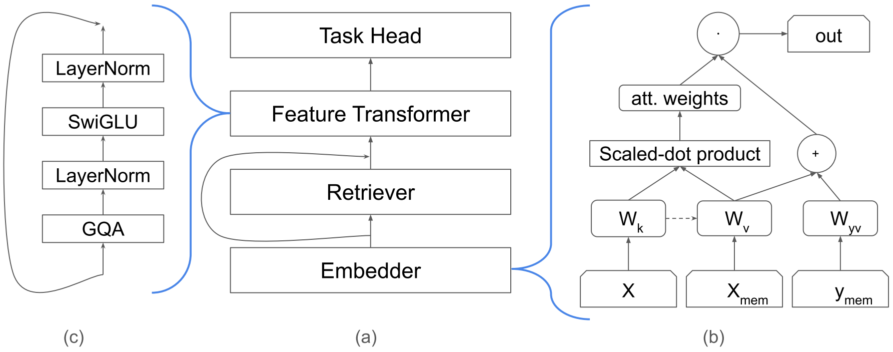

Fig. 1: Outline of the RAM architecture. (a) On a high level, RAM consists of an Embedder, Retriever, Feature Transformer and a Task-specific head. (b) The core of the embedder is a modified cross-attention layer that mixes the true targets of memory samples into the value projection. (c) The feature transformer base block consists of Grouped Query Attention followed by SwiGLU.

also employ the weight-sharing between blocks. This behaviour is controlled by two hyperparameters n_blocks and $n\_repeats$, where n_blocks defines the number of blocks with unique weights and $n\_blocks \cdot n\_repeats$ defines the total number of blocks.

Task specific head is responsible for producing the final outputs for downstream tasks. For both classification and regression we use a global average pooling across the feature dimension followed by a single linear layer that processes the pooled representation.

3.1 Training Procedure and Domain-Aligned Memory

To improve the RAM's ability for in-context domain adaptation, we introduce a Domain-Aligned Memory (DAM) training procedure. This method operates under the assumption that the dataset consists of observations from multiple distinct domains. Before training, we partition the dataset and the labelled memory based on the domain, and during training operate on a single subset at a time. This way we ensure that the domains of both the input and the memory are always aligned during training and the model is able to concentrate on domain-specific patterns.

4 Experiments and Results

To evaluate the effectiveness of RAM for in-context domain adaptation, we conducted experiments to measure its performance under domain shifts. Given the limited availability of multi-domain tabular datasets, we resorted to synthetic datasets which were generated in three steps. First, we randomly sampled domains (hidden features) and a set of scenarios (observable features) for each domain from a normal distribution. Afterwards, the labels were obtained by passing the concatenated samples through a randomly initialized non-linear neural network. Finally, the hidden features were discarded, retaining only a domain-identifier needed for DAM grouping.

We compared the performance of RAM with traditional machine learning models: Linear Regression, K-Nearest Neighbors (KNN), and Random Forest. For RAM, we used default hyperparameters across all experiments due to prohibitive computational costs for performing an extensive hyperparameter optimization. For the baseline models we used grid search on a dedicated validation subset of domains.

As shown in Table 1, RAM outperformed all other tested methods, indicating that it can utilise the provided memory to infer the characteristics of the target domain. To further investigate the importance of the retrieval block, we conducted an ablation study where (a) only labels were used as values in the attention layer, and (b) only features were used as values. While using only labels resulted in slightly worse performance, excluding the labels led to a major drop in performance. Additionally, to assess whether RAM truly utilizes its pre-training across multiple domains, or merely acts as a KNN, we compared its performance with a set of per-domain k-nearest regression models trained directly on the memory. Even in this setting, RAM was a better performing method, further reinforcing the assumption that RAM's architecture and training procedure allows it to achieve superior generalization.

Model	MSE	R^2
RAM	1.89 ± 0.21	0.93 ± 0.01
RAM (y only)	4.06 ± 0.49	0.86 ± 0.01
RAM (X only)	7.87 ± 1.15	0.67 ± 0.08
Linear Regression	13.29 ± 0.99	0.46 ± 0.08
Random Forest	52.67 ± 2.73	-0.99 ± 0.10
KNN	7.68 ± 1.08	0.67 ± 0.06
KNN (retrained on domain memory)	8.22 ± 0.65	0.69 ± 0.02

Table 1: Comparison of RAM with the baselines.

5 Conclusion

RAM demonstrates promising results in in-context domain adaptation for tabular data, outperforming traditional machine learning models in synthetic domain-shift scenarios. The architecture is able to effectively use its retrieval mechanisms and the Domain-Aligned Memory training procedure. However, to fully validate RAM's robustness, future work should focus on testing the model in real-world settings with diverse and complex domains.

References

[1] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[2] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[4] Bernhard Schäfl, Lukas Gruber, Angela Bitto-Nemling, and Sepp Hochreiter. Hopular: Modern hopfield networks for tabular data. *arXiv preprint arXiv:2206.00664*, 2022.

[5] Jannik Kossen, Neil Band, Clare Lyle, Aidan N Gomez, Thomas Rainforth, and Yarin Gal. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. *Advances in Neural Information Processing Systems*, 34:28742–28756, 2021.

[6] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342*, 2021.

[7] Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. Tabr: Unlocking the power of retrieval-augmented tabular deep learning. *arXiv preprint arXiv:2307.14338*, 2023.

[8] Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

[9] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[10] Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *arXiv preprint arXiv:2402.14905*, 2024.