Predictive and Explainable Analysis of Post-operative Acute Kidney Injury in Children undergoing Cardiopulmonary Bypass: An Application of Large Language Models

Mansour T.A. Sharabiani[1[0000−0003−3808−277X]], Alireza S. Mahani[2[0000−0002−7932−6681]], Alex Bottle[1[0000−0001−9978−2011]], Yadav Srinivasan[3[0009−0003−1897−3548]], Richard Issitt[4,5,6[0000−0002−6299−2557]], and Serban Stoica[7[0000−0002−0456−1249]]

[1] School of Public Health, Imperial College London, UK
[2] Quantitative Research, Davidson Kempner Capital Management, New York, US
[3] Cardiac Surgery Dept, Great Osmond Street Hospital for Children, London, UK
[4] Centre for Heart Failure, Transplantation and Extracorporeal Support, UCL Institute for Cardiovascular Science, London, UK
[5] Perfusion Department, Great Ormond Street Hospital for Children, London, UK
[6] Data Research, Innovation and Virtual Environments Unit, Great Ormond Street Hospital for Children, London, UK
[7] Cardiac Surgery Department, Bristol Royal Children's Hospital, Bristol, UK

Abstract. Motivation: Embedding large language models (LLMs) hold promise for predictive analytics due to their natively consistent numeric output. This study explores the utility of general-purpose LLMs for extracting interpretable and actionable insights from electronic health record (EHR) text columns in tabular data, focusing on paediatric cardiopulmonary bypass (CPB).
Methods: We analysed data from 963 paediatric CPB operations in the UK (2019-2021), focusing on the severity of post-operative acute kidney injury (AKI) as a binary outcome, and using text columns documenting planned surgical procedures and patient diagnoses for each operation as features. We employed OpenAI's `text-embedding-3-large` for embeddings and `gpt-4-turbo` for generating descriptive labels of patient clusters that were formed by applying spherical k-means to embedding vectors. These 'AI clusters' were compared against the 'expert clusters' - based on the Partial Risk Adjustment in Surgery (PRAiS) v2 protocol - for 1) consistency using adjusted rand index (ARI) between the clusters and 2) predictive performance using area under ROC (AUC) of logistic regression models using cluster memberships as a categorical variable.
Results: 'AI clusters' showed statistically significant levels of consistency with 'expert clusters', evidenced by their ARIs of 0.31 and 0.32 for planned procedures and diagnoses, respectively. The clusters demonstrated comparable discriminative power in predicting severe post-operative kidney injury, with AUCs of 0.63 vs. 0.60 (planned procedures) and 0.56 vs. 0.58 (diagnoses) for AI clusters vs. expert clusters. Notably, AI clusters identified three interventions with significant odds ratios for AKI severity, highlighting potential areas for clinical focus. Replacing k-means clustering followed by logistic regression with k-nearest-neighbours - applied to LLM text embeddings - further improved AUC to 0.66 and 0.63, for planned procedures and diagnoses, respectively.
Conclusions: Our findings affirm the potential of LLMs as effective tools in medical text analysis, facilitating both exploratory and predictive tasks. The integration of LLM-derived insights with traditional data analysis methods can significantly enhance risk stratification and outcome prediction in paediatric CPB, underscoring the value of AI-driven approaches in complex healthcare datasets.

Keywords: generative artificial intelligence · electronic health records · cardiopulmonary bypass

1 Introduction

1.1 Text embedding and modern LLMs

The impressive capabilities of text-completion large language models (LLMs), perhaps best exemplified by OpenAI's ChatGPT, have dominated much of the 'GenAI' headlines. Yet, in predictive analytics, e.g., when the objective is to predict the risk of a future health event, 'embedding' LLMs are arguably more promising, thanks to their natively-numeric (rather than text) output which can be readily used by machine learning

algorithms. Despite their distinct output types, embedding LLMs are deeply connected to text-completion LLMs, as they are often the result of extracting the first few layers of a text-completion LLM and fine-tuning it on tasks specifically designed for embedding models. This is an example of 'transfer learning'.

But what is text embedding? Simply put, it means converting text - a single word or a sequence of words forming a sentence, paragraph, or an entire article - to a numeric vector - often high-dimensional - such that semantic relationships and similarities in the input space (natural language) are preserved in the output space (numeric vectors). Since embedding a text string produces a vector, embedding a text 'column' results in a 'matrix' (Figure 1). Each column of this matrix can be included as a feature in a predictive model.

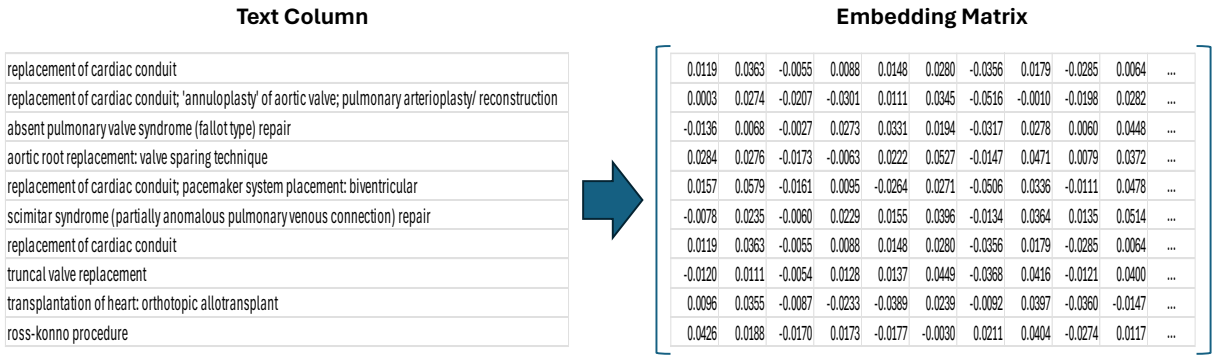

Fig. 1. Illustration of the text embedding concept. Right: A text column representing the planned CPB procedures for 10 pediatric patients. Right: The embedding matrix produced by OpenAI's `text-embedding-3-large`, with each patient represented by a high-dimensional, numeric vector. The embedding algorithm maps semantically-similar text to close-by points in the numeric embedding space. While here only 10 columns of the embedding matrix are shown, modern embedding LLMs produce much higher-dimensional vectors. Embeddings vectors are often pre-normalised, i.e., the L2 norm of each row equals 1.

The evolution of text embedding algorithms can be summarised in three generations:

- Generation 1: Sparse representations with no semantic information. Examples: one-hot encoding, bag-of-words.
- Generation 2: Some semantic information captured, but little context used for interpreting the words. Examples: Latent Semantic Analysis (LSA), Word2Vec/Doc2Vec, GloVe.
- Generation 3: Transformer-based models that create deep contextualized word embeddings. Examples: BERT, GPT.

In this paper, we include an example from each of the above categories (see Methods), and compare their performance.

1.2 Related Work

Despite the natural appeal of embedding LLMs for predictive models due to their numeric output, this potential appears to be under-explored. For instance, [4] conducted a systematic review of research on applying natural language processing (NLP) to electronic health records (EHR). They find that most use cases considered are in medical note classification, clinical entity recognition, text summarisation, deep learning (DL) and transfer learning architecture, information extraction, and medical language translation. In particular, they do not report on any research applying embedding to text columns based on a patient's EHR.

In [11], authors apply Med-BERT to two disease prediction tasks: 1) prediction of heart failure among patients with diabetes and, 2) prediction of onset of pancreatic cancer. However, they use detailed, longitudinal EHR, access to which is not feasible in many settings. Also, they pretrain a BERT on EHR data,

which requires significant domain expertise in AI and neural networks. Finally, while they report impressive performance gain from Med-BERT, yet the model and data are not publicly available, due to privacy concerns.

1.3 Our Contributions

The overarching goal of this research is to facilitate the adoption of LLMs in predictive problems, especially in healthcare. In particular, our target audience is data scientists working in the healthcare field, without any extensive prior experience/knowledge in AI, deep learning, and large language models. To this end, we identify and address three challenges:

1. *Scepticism towards the usefulness of general-purpose LLMs in specialised domains* - We present empirical evidence supporting the medical plausibility of embedding LLM output including a) Consistency of AI and domain experts in grouping patients according to EHR text, and b) Comparable performance of AI and expert clusters in predicting severe postoperative AKI.
2. *Lack of interpretability of text embedding vectors* - We propose and demonstrate a workflow for using text-completion LLMs for interpreting the output of embedding LLMs. Also, co-authors with clinical expertise reviewed and validated the LLM interpretations for our paediatric CPB case study.
3. *Technical barriers for data scientists without deep AI expertise* - We have created and published a free, open-source implementation - in Python - of all steps in our explanatory and predictive workflows, along with a detailed tutorial.

It is also worth noting that our choice of dataset/problem (see Methods) gave us two advantages:

1. The availability of PRAiS protocols for grouping patients according to their planned procedures and diagnoses created reference points for assessing the output of embedding LLMs applied to the same two text fields, especially their medical plausibility.
2. Since each value in the *planproc* and *diagnosis* text fields consists of one or more standardised medical codes (Figure 1, left side), we are able to use a so-called 'Bag-of-Codes' (BoC) encoding for each one (see Methods). By comparing the performance of BoC against Doc2Vec and OpenAI's LLM, we are able to isolate and quantify the contribution of semantics information towards the quality of text embeddings in our problem.

2 Methods

2.1 Data

Data for 963 consecutive, single-centre paediatric CPB operations in the UK during 2019-2021 were used for this analysis. Two EHR text fields were considered for embedding: planned surgical procedures (*planproc*) and patient diagnoses (*diagnosis*). We also mapped patients to a smaller number of clusters, using the PRAiS protocol [7] for each of the two fields. For more details on how the data was collected and prepared, see [15].

2.2 Computing Environment

We used a combination of R [9] (version 4.2.2), Python [8] (version 3.11.9) and their various packages, running on a PC with Intel Core i7 and 16GB of RAM, and a 64-bit Windows 11.

2.3 Text Embedding Methods

We evaluated three algorithms, each representing a generation of text embedding techniques (see Introduction):

1. Bag-of-Codes (BoC) (Gen 1), implemented in R by the authors. Each medical code gets a binary indicator, and each instance of a text field gets 1's for all medical codes present in the string, and 0's for others. The output dimension was dictated by the number of distinct medical codes used for each field, and was 177 and 282 for *planproc* and *diagnosis*, respectively.

2. Doc2Vec (Gen 2) [6], acccessed via the `doc2vec` R package. We use the 'Distributed Bag-of-Words' (DBOW) variation with output dimension set to 100 in all our experiments.
3. OpenAI's `text-embedding-3-large` (Gen 3), accessed via the `openai` R package [12]. It outputs high-dimensional embedding vectors (length 3,072).

2.4 Spherical K-means Clustering

We use 'spherical' k-means, accessed via the `skmeans` R package [3], for clustering patients according to their text embedding vectors. This is a variant of standard k-means, where the metric used to measure the distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ of length N is the 'cosine distance', $D_C(\mathbf{x}, \mathbf{y})$:

$$D_C(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^\mathbf{T}\mathbf{y}}{(\mathbf{x}^\mathbf{T}\mathbf{x})^{1/2}\,(\mathbf{y}^\mathbf{T}\mathbf{y})^{1/2}} \tag{1}$$

This metric is invariant with respect to the length of the two vectors, focusing only on their angles and thus making it a suitable choice for text embedding vectors that are L2-normalised, i.e., $\mathbf{x}^\mathbf{T}\mathbf{x} = 1$.

2.5 Cluster Similarity

We use Adjusted Rand Index (ARI) [10] to quantify the similarity of two data clusterings. The Rand Index between clusterings A and B reflects the probability that a pair of observations that belong to the same partition according to clustering A also belong to the same partition in clustering B. The 'adjusted' Rand Index subtracts a baseline probability that reflects the similarity occurring by chance. We use the `R` package `mclust` for calculating ARI [14].

2.6 Predictive Performance of Embeddings

To compare the predictive performance of embeddings, we take two approaches. First, we use the result of clustering applied to embedding vectors. Specifically, we train a logistic regression model with cluster membership vector of observations as the sole, categorical variable. This is equivalent to using the mean outcome probability of observations falling within each cluster as the predicted probability for all of those observations.

Secondly, we use a flexible version of the above approach by using k-nearest-neighbours. Rather than a data point being assigned to a predetermined set of clusters, it gets the mean outcome probability associated with its nearest neighbours, with number of neighbours being a hyperparameter. Note that, for pre-normalised embedding vectors, using an L2 distance metric in knn is mathematically equivalent to the cosine distance metric.

2.7 LLM Interpretation of Clusters

We concatenate the *planproc* and *diagnosis* fields for each patient, and then present this concatenated field for all patients, grouped into the clusters returned by applying spherical k-means to the embedding vectors, to OpenAI's long-context (128k) text-generating LLM, `gpt-4-0125-preview`, and solicit representative and distinct labels for the clusters. Figure 2 shows an abbreviated version of the submitted prompt (left), and the response (right).

2.8 Embedding-based Risk Score

We want to generate a single risk score/variable to summarise the information extracted from a text field via embeddings, in a numeric form that can be included alongside other variables in a predictive model. This is generally preferred, rather than including the raw embedding vectors since these vectors are high-dimensional, e.g. 3072 for OpenAI's `text-embedding-3-large`.

To make sure the value of the risk score for each patient has not been impacted by the outcome variable for that patient (otherwise, the final model would suffer from overfitting), we use a cross-validation approach similar to what is done in stacked generalisation [16].

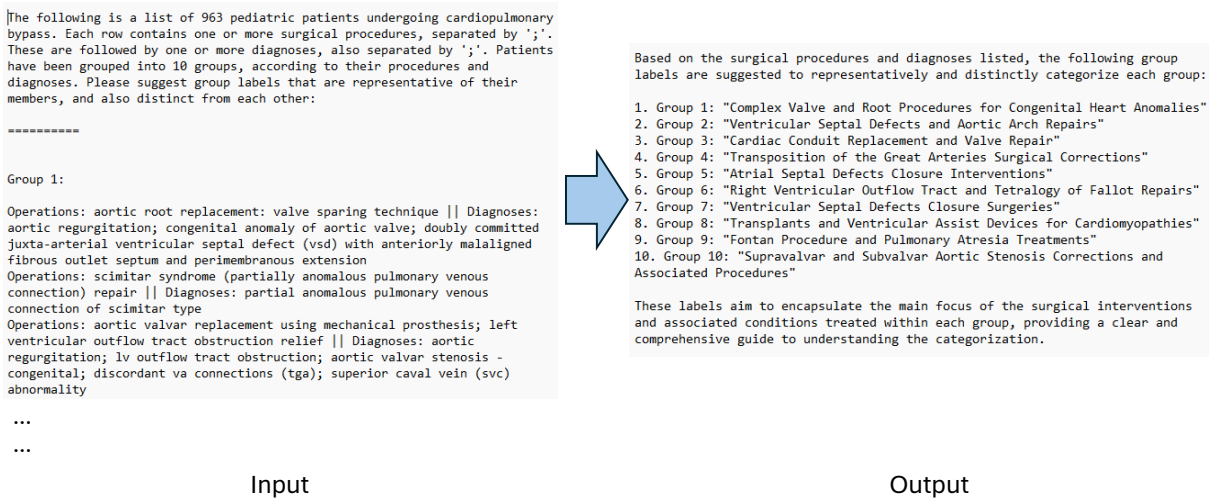

Fig. 2. AI interpreting AI. Input (left): The prompt submitted to `gpt-4-turbo`, a long-context model from OpenAI. Clusters were formed by applying spherical k-means to the combined LLM-embedding vectors of *planproc* and *diagnosis*. The prompt is truncated for brevity. Output (right): The response from `gpt-4-turbo`, suggesting medically plausible labels for patient clusters.

For the underlying supervised learning algorithm used to generate the risk score, we use the k-nearest-neighbour algorithm, similar to before. However, here we use knn only on the embedding vectors, and then include the predicted risk score alongside a set of baseline variables (e.g., patient age and gender) in a final, logistic regression model.

3 Results

In the first two subsections, we present evidence to support the medical plausibility of LLM embeddings. We then focus on the predictive performance of embedding vectors.

3.1 AI Clusters vs Expert Clusters

Figure 3 shows the consistency - measured via ARI - of clusters based on three text embedding algorithms ('AI clusters' for short) with expert clusters. We observe that:

- AI clusters are consistent with expert clusters, reflected in positive ARI numbers for *planproc* and *diagnosis*. In other words, all text embedding algorithms are successful at extracting domain knowledge from medical text in a way that is in line with domain experts.
- This consistency is higher when using a 3rd-gen algorithm (OpenAI's LLM), compared with earlier-generation embedding techniques such as Doc2Vec and BoC. This suggests that the transformer-based LLM has been successful at extracting useful contextual information from multiple medical codes in patients' *planproc* and *diagnosis* text strings.

3.2 AI Explaining AI

To interpret the patient groups created from embeddings of *planproc* and *diagnosis*, we use OpenAI's long-context, `gpt-4-turbo` model to produce descriptive labels for each group. Figure 2 shows the input and output, both truncated due to space constraints.

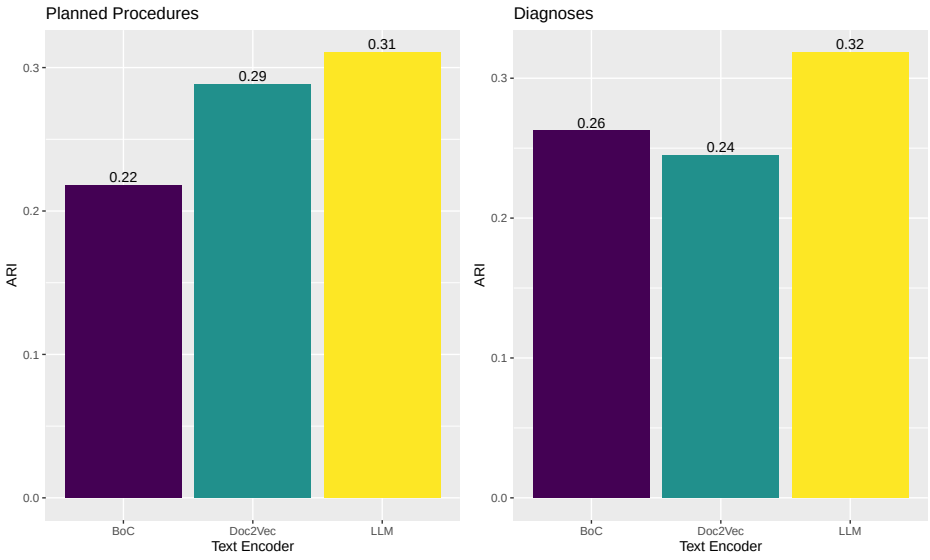

Fig. 3. Consistency - measured by ARI - between AI clusters and expert clusters for various text encoding algorithms, and for two text columns, *planproc* (left) and *diagnosis* (right). BoC = Bag of Codes; LLM stands for OpenAI's `text-embedding-3-large` model. AI clusters are the result of applying spherical k-means - with 10 clusters and 10 random starts - to text embeddings.

Table 1 shows the full table of cluster labels, along with their odds-ratio and p-value, where the outcome variable is severe post-operative AKI. Three groups stand out, with the first two having lower than average risks, and the 3rd one having higher than average risk: 1) 'Cardiac Conduit Replacement and Valve Repair' (OR = 0.37, p-val = 0.01), 2) 'Arterial Septal Defects Closure Interventions' (OR = 0.25, p-val $<$ 0.01) and 3) 'Transplants and Ventricular Assist Devices for Cardiomyopathies' (OR = 5.9, p-val $<$ 0.01). A subsequent review by clinical experts (co-authors of the paper) validated the AI findings: 1) ASD patients have a shorter bypass time, which would explain the lower risk of post-operative AKI; 2) Transplant/VSD patients have significant comorbidity in terms of long-term low cardiac output; 3) The conduit group would be those that potentially have beating heart surgery.

3.3 Predictive Value of LLM Embeddings

Figure 4 compares the accuracy of predicting severe AKI (measured as area under ROC) using *planproc* or *diagnosis* fields, and applying different encoding techniques (BoC / Doc2Vec / LLM) and different supervised learning algorithms (K-means + logistic regression / KNN-50). Observations:

- KNN applied to LLM embeddings outperforms expert clusters + logistic regression (dotted lines). (Note that when we use clustering - whether expert clusters or k-means applied to text embeddings - for feature extraction and get back cluster membership as a categorical feature, logistic regression on this feature is the main viable supervised learning approach, and is equivalent to using the average value of binary outcome within each cluster as probability prediction for that cluster.)
- For both text fields and all three embedding approaches, KNN outperform clustering followed by logistic regression. This is plausible since KNN can be thought of as dynamic clustering, adjusting the cluster memberships for each point in the feature space. The ability to use KNN with embeddings thus provides an advantage over using expert clusters.
- Performance improves with newer-generation encoding algorithms (LLM $>$ Doc2Vec $>$ BoC). This suggests that a modern general-purpose LLM is successful at extracting semantic and contextual information from medical text that is informative for predicting post-operative AKI.

Table 1. AI explaining AI - Group descriptions are generated by the `gpt-4-turbo` model, using patient clusters that are based on combined embeddings of *planproc* and *diagnosis* by the `text-embedding-3-large` model. 'Severe' and 'Total' represent the number of patients with severe AKI and total patients, respectively, within each cluster. OR stands for odds ratio of severe AKI for that cluster compared to the remaining patients. P-value is based on a two-sided Fisher test. Rows in green highlight patient clusters with lower risk of severe AKI than others, while the row in red represents patients with higher risk than others.

Group No.	**Group Description (AI generated)**	**Severe/Total (%)**	**OR**	**P-val**
1	Complex Valve and Root Procedures for Congenital Heart Anomalies	39/153 (25.5%)	0.84	0.84
2	Ventricular Septal Defects and Aortic Arch Repairs	29/92 (31.5%)	1.33	0.26
3	Cardiac Conduit Replacement and Valve Repair	7/57 (12.3%)	0.37	0.01
4	Transposition of the Great Arteries Surgical Corrections	8/41 (19.5%)	0.67	0.37
5	Atrial Septal Defects Closure Interventions	7/79 (8.9%)	0.25	8e-5
6	Right Ventricular Outflow Tract and Tetralogy of Fallot Repairs	33/102 (32.4%)	1.39	0.15
7	Ventricular Septal Defects Closure Surgeries	41/147 (27.9%)	1.10	0.68
8	Transplants and Ventricular Assist Devices for Cardiomyopathies	30/46 (65.2%)	5.88	1.4e-8
9	Fontan Procedure and Pulmonary Atresia Treatments	19/78 (24.4%)	0.89	0.79
10	Supravalvar and Subvalvar Aortic Stenosis Corrections and Associated Procedures	12/59 (20.3%)	0.70	0.36

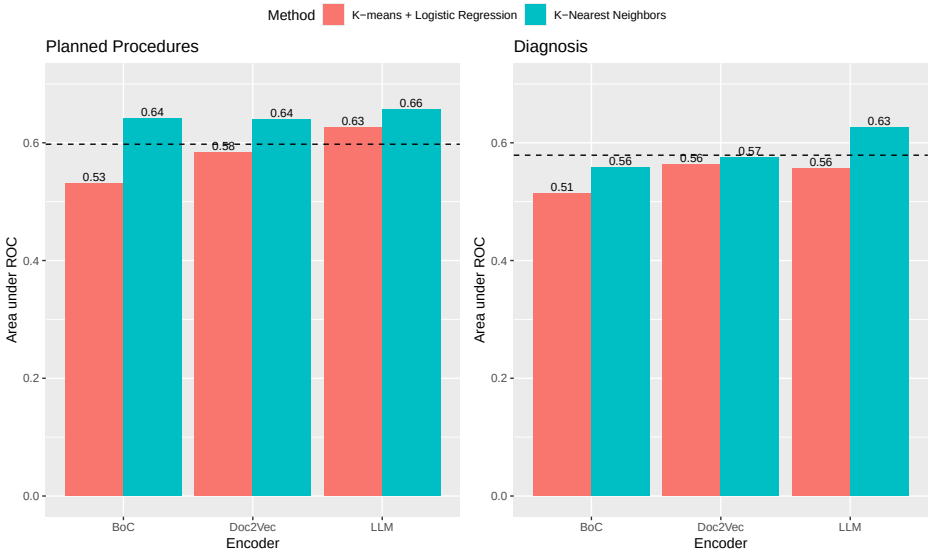

Fig. 4. Comparison of area under ROC for predicting AKI severity, using *planproc* (left) and *diagnosis* (right) text fields, and applying three encoding algorithms. BoC = Bag of Codes, LLM: OpenAI's `text-embedding-3-large` model. Red bars: Applying spherical k-means to the embedding matrix, followed by logistic regression on cluster memberships as a categorical feature. Green bars: KNN (50 neighbours) applied directly to the embedding matrix. Area under ROC is the aggregate performance over 10-times repeated, 5-fold cross-validation. Dotted, horizontal line represents performance of logistic regression using expert-cluster (PRAiS v2 protocol) memberships as a categorical feature.

3.4 Creating a Synthetic Risk Score from Medical Text

We saw that KNN - as a flexible alternative to clustering plus logistic regression - can be applied to text embeddings with good predictive performance. In real-world predictive problems, we have other variables besides text fields. For instance, in our problem of predicting severe AKI, we may want to use basic patient attributes such as gender, age, height, and weight. We may prefer to use other supervised algorithms for our final model, including linear ones, perhaps due to easier interpretation. Therefore, we propose a two-stage approach for incorporating text fields into the final model:

1. After embedding the text field, distil it into a single variable by training a KNN model on the embedding vectors, and using the predictions of the KNN model as a synthetic variable or risk factor. To avoid overfitting, wrap this process in a cross-validation scheme.
2. Add the above, synthetic risk score to other variables and build the final model.

Figure 5 illustrates the results of applying the above process to our data, using two sets of variables as baseline. In first setting (red lines), baseline variables are patient's sex, age, height and weight at the time of operation. In the second settings (blue lines), in addition to the first set of variables, we have also included operation time. We report the AUC improvement vs. using no text embeddings (y axis), while using different lengths of the embedding vector (x axis) in training the KNN model. Observations:

- Improvements due to the synthetic risk score (both *planproc* and *diagnosis*) are significantly smaller after adding the operation time as a feature. This suggests that there is significant correlation between patients' diagnoses, their planned procedures, and the length of the operation. This is medically quite plausible.
- At the same time, even after adding operation time to baseline variables, there is still measurable benefit from adding the text-embedding-based feature (up to about 2%).
- AUC improvement is sensitive to the number of embeddings used in training the KNN model, with performance flattening or degrading after about 30 embeddings. This is likely due to the negative effect of overfitting outweighing the benefits of including more information in the KNN model.

4 Discussion

4.1 Summary

We presented empirical evidence illustrating the ability of publicly-available, general-purpose large language models to extract information from medical text columns in tabular datasets that is consistent with domain experts and improves the accuracy of predictive models. To streamline this information extraction, we proposed an explanatory recipe (Algorithm 1) and a predictive recipe (Algorithm 2) that suit the normalised and high-dimensional nature of text embedding vectors produced by LLMs, and minimise the risk of overfitting in small datasets.

All steps described in our recipes have been implemented in the `TextEmbeddingFE` Python package, available on github. The repository also includes a step-by-step tutorial for how to use the package. We hope that these recipes and their software implementation make it easier for practitioners to use LLMs in predictive problems in healthcare.

4.2 Future Work

In our proposed explanatory workflow (Algorithm 1), we used a particular clustering algorithm, i.e., spherical k-means. An area of future research to identify and explore other suitable clustering algorithms that work well with data laying on a hypersphere. Examples include DBSCAN [13], mean shift clustering [1] and affinity propagation [2]. Similarly, we used k-nearest-neighbour as the supervised learning method for creating the synthetic score from embedding vectors. In future research, we explore alternatives to knn such as learned vector quantisation and self-organising maps [5].

The two workflows also included a few hyperparameters, whose values we set based on heuristics and judgement. Examples include the number of embedding-vector elements used in clustering/KNN and the

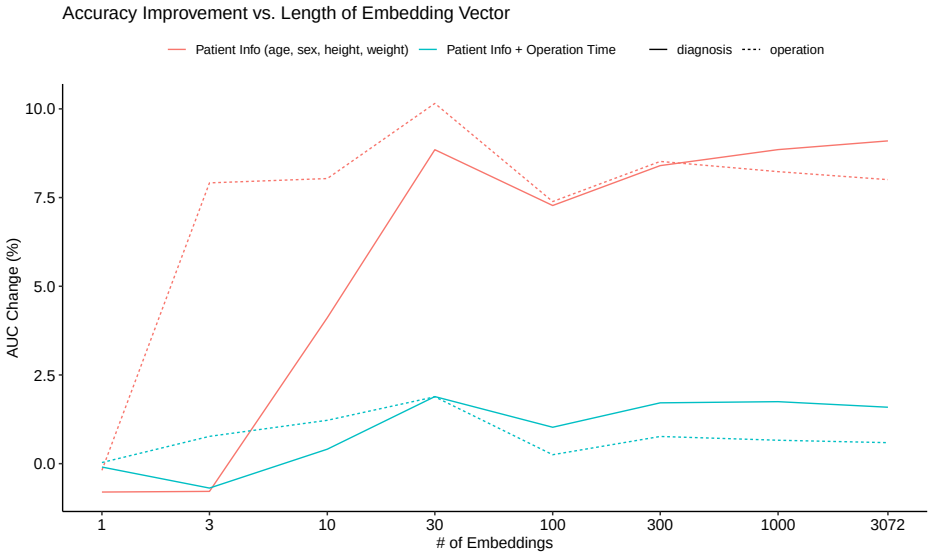

Fig. 5. Improvement in area under ROC (AUC) - due to adding text-embedding-based synthetic risk score as a feature - for models predicting severe post-operative AKI. Red lines correspond to models that include only patient attributes at time of operation (age, sex, height, weight) as baseline variables. Blue lines include patient attributes plus operation time. Solid lines correspond to synthetic risk score being derived from embedding of the *diagnosis* column, while dashed lines correspond to the *planproc* column. AUC change is based on 10-times repeated 5-fold cross-validation. In generating the synthetic risk score, 10-fold cross-validation was applied to k-nearest-neighbours with 50 neighbours.

Algorithm 1 Proposed workflow for using LLMs to extract insight from a text column in tabular data with a binary outcome variable. `Text` in parentheses refer to function(s) in the `TextEmbeddingFE` Python package that perform each step.

1. Convert text column to numeric matrix using embedding LLM (`embed_text`).
2. Cluster observations by applying spherical k-means to the embedding matrix (`cluster_embeddings`).
3. Submit clustered observations (their text field) to text-completion LLM to generate cluster summaries/labels (`generate_prompt`, `interpret_clusters`).
4. Perform Fisher test on crosstab of cluster membership vector and response vector to identify clusters with significantly higher/lower probability of positive response (`fisher_test_wrapper`).
5. Combine steps 3 and 4 to describe groups of observations with significant odds-ratio vs. rest of population.

Algorithm 2 Proposed steps for using LLMs to incorporate a text field into a predictive model. `Text` in parentheses refer to function(s) in the `TextEmbeddingFE` Python package that perform each step.

1. Convert text column to numeric matrix using embedding LLM (`embed_text`).
2. Split the data into a training set (`X_train`, `y_train`) and a test set (`X_test`, `y_test`).
3. Fit a cross-validated k-nearest-neighbour model in the training set (`FeatureExtractor_BinaryClassifier.fit(X_train, y_train)`).
4. Use predicted probabilities from model trained in step 3 to generate a synthetic feature for training (`FeatureExtractor_BinaryClassifier.predict()`) and test (`FeatureExtractor_BinaryClassifier.predict(X_test)`) sets.
5. Add the synthetic feature from step 4 to other features to build the final predictive model.

number of neighbors used in KNN. In predictive modelling, in particular, tuning these hyperparameters may be important since their values may drive the performance of the model. In future releases of `TextEmbeddingFE`, we will consider including facilities for performing hyperparameters tuning, ideally as part of the hyperparameter tuning for the final predictive model.

Disclosure of Interests. The authors have no competing interests to disclose.

References

1. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE transactions on pattern analysis and machine intelligence **17**(8), 790–799 (1995)
2. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. science **315**(5814), 972–976 (2007)
3. Hornik, K., Feinerer, I., Kober, M., Buchta, C.: Spherical k-means clustering. Journal of statistical software **50**, 1–22 (2012)
4. Hossain, E., Rana, R., Higgins, N., Soar, J., Barua, P.D., Pisani, A.R., et al.: Natural language processing in electronic health records in relation to healthcare decision-making: a systematic review. arXiv preprint arXiv:2306.12834 (2023)
5. Kohonen, T., Kohonen, T.: Learning vector quantization. Self-organizing maps pp. 245–261 (2001)
6. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196. PMLR (2014)
7. Pagel, C., Rogers, L., Brown, K., Ambler, G., Anderson, D., Barron, D., Blackshaw, E., Crowe, S., English, K., Franklin, R., et al.: Improving risk adjustment in the prais (partial risk adjustment in surgery) model for mortality after paediatric cardiac surgery and improving public understanding of its use in monitoring outcomes. Health Services and Delivery Research **5**(23), 1–164 (2017)
8. Python Core Team: Python: A dynamic, open source programming language. `https://www.python.org` (2023), python version 3.11.9
9. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2021), `https://www.R-project.org/`
10. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association **66**(336), 846–850 (1971)
11. Rasmy, L., Xiang, Y., Xie, Z., Tao, C., Zhi, D.: Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. NPJ digital medicine **4**(1), 86 (2021)
12. Rudnytskyi, I.: openai: R Wrapper for OpenAI API (2023), https://github.com/irudnyts/openai, https://irudnyts.github.io/openai/
13. Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X.: Dbscan revisited, revisited: why and how you should (still) use dbscan. ACM Transactions on Database Systems (TODS) **42**(3), 1–21 (2017)
14. Scrucca, L., Fraley, C., Murphy, T.B., Raftery, A.E.: Model-Based Clustering, Classification, and Density Estimation Using mclust in R. Chapman and Hall/CRC (2023). `https://doi.org/10.1201/9781003277965`, `https://mclust-org.github.io/book/`
15. Sharabiani, M.T.A., Issitt, R.W., Mahani, A.S., Srinivasan, Y., Stoica, S.: A dynamic time-series model of oxygen consumption during paediatric cardiopulmonary bypass. medRxiv pp. 2024–03 (2024)
16. Wolpert, D.H.: Stacked generalization. Neural networks **5**(2), 241–259 (1992)

A Acronyms

- AKI: acute kidney injury
- AUC: Area under ROC
- API: Application Programming Interface
- BERT: Bidirectional Encoder Representations from Transformers
- CPB: cardiopulmonary bypass
- EHR: Electronic Health Records
- ERM: Embedding-based Risk Model
- FRM: Final Risk Model
- GenAI: Generative Artificial Intelligence
- GTE: Generalised Target Encoding
- GTE-CV: Generalised target encoding using cross-validation
- GPT: Generative Pre-trained Transformer
- KDIGO: Kidney Disease Improving Global Outcomes
- LLM: large language model
- ML: Machine Learning
- MVST: Multi-valued Standardised Text
- NER: Named Entity Recognition
- NLP: Natural Language Processing
- PC: Principal Component
- PCA: Principal Components Analysis
- TF-IDF: Term-Frequency, Inverse Document Frequency