Kalpa Publications in Computing
Volume 22, 2025, Pages 732–743
Proceedings of The Sixth International Conference on Civil and Building Engineering Informatics

Leveraging Transformers for Improved Decision-Making in Road Maintenance

Rui Kang[1]*, Stephen Green[2], and Ioannis Brilakis[3]

[1] University of Cambridge, Cambridge, U.K.
rk703@cam.ac.uk
[2] University of Cambridge, Cambridge, U.K.
slg79@cam.ac.uk
[3] University of Cambridge, Cambridge, U.K.
ib340@cam.ac.uk

Abstract

Transportation performance is heavily influenced by the overall quality and the effectiveness of road maintenance. However, this remains an expert-dependent activity, despite recent efforts to digitalize road geometries and management processes. Road maintenance knowledge accumulated through addressing relevant enquiries is inexplicitly learned by experts and transferred into experience, which contributes very little to developing maintenance digitalization techniques and automated decision-making processes. In this case, fully utilizing historical maintenance records and turning them into computer-readable knowledge is a crucial task to be solved. This paper aims to extract key information from road maintenance request texts and then implement step-by-step thinking to make road maintenance decisions. This chain of thought is first proposed by reviewing the key elements and logical flow of road maintenance decision-making. Then, a cross-attention mechanism based on a transformer architecture is implemented on maintenance record texts and target knowledge element sequences. The result of this experiment overperforms on a pre-trained BERT model and demonstrates a valid performance on the text-knowledge alignment in road maintenance domain. The method proposed in this paper provides a solution for reliable and traceable decision-making and shows a promising application in domain-specific knowledge management.

Contents

*Corresponding author

1 Introduction

Roads are the main mode of transport for individuals and their maintenance has a great impact on the service quality of roads across the life cycle. For example, over 92% of the distance travelled by passengers in the UK is covered by roads, accounting for 595 billion passenger kilometers[7]. Over 60% of motorways in the UK were built during the 1960s and 1970s [10]. These aging road networks are facing problems caused by terrible road conditions. In England, about 39% of roads are marked with visible deterioration or in urgent need of maintenance [1]. Untimely maintenance will further increase costs and cause rapid deterioration of road assets. Hence, effective and reliable decision-making can have an important role in road maintenance management.

Road maintenance management processes can be divided into four types: planning, programming, preparation and operation [16]. Table 1 shows the spatial coverage, time horizons and staff involved in these processes. For the former two processes, long and medium-term systematic planning over a year-long is conducted by professionals taking policy and budgets into account. For the latter two processes, cyclical and reactive maintenance decisions are made by engineers and technicians to respond to regular inspections and reports. High-quality road maintenance decision-making is required to minimize traffic disruption, prevent safety issues, and improve cost effectiveness. However, decision-making in road maintenance remains heavily expert dependent. Despite the use of data management systems to record road maintenance activities, there are no effective measurements that use these records to support or automate decision-making processes. These decisions involve identifying road assets and defects, determining appropriate repair jobs, and prioritizing jobs based on asset importance, defect severity and job urgency. Such decisions still primarily rely on the accumulated experience of engineers.

Management process	Spatial coverage	Time horizon	Staff concerned
Planning	Network-wide	Long term	Policy level professional
Programming	Network to sub-network	Medium term	Middle level professional
Preparation	Section or project	Budget year	Engineer/technician
Operation	Sub-section or activity	Actual duration	Technician/worker

Table 1: Road maintenance management process type

Rule-based methods are commonly used to efficiently make use of human expertise in road maintenance decision-making. One of the most straightforward techniques is a decision tree. Since road defects are finite, engineers can manually construct decision trees to select appropriate treatments. There are also studies that use decision trees to select maintenance activities and support complicated decision-making processes [2, 8]. Since the capacity of addressing complex problems is limited by the simplicity of decision trees, expert systems are proposed to resolve intricate decision-making problems. Tailored by experts, expert systems compile and formalize expert knowledge into rules and use an inference engine to handle knowledge [19]. Compared to expert decisions, expert systems provide competent solutions for material selection[17] and task prioritization [13] in the road maintenance domain. However, since rule-based methods are encoded from human knowledge and maintained by domain experts, they lack the capability to learn from historical data and flexibility to deal with any unforeseen changes.

Natural Language Processing (NLP) methods and Large Language Models (LLMs) are regarded as promising solutions to manage knowledge and support decision-making. [6] investigated how machine learning methods are integrated in NLP and the automation the priority assignment of building maintenance tasks. This work verified the effectiveness of NLP in conducting a preliminary check of the most urgent requests. Similarly, NLP is used to rank the severity of maintenance requests through text mining and to support maintenance decision-making [5]. With the development of LLMs, pretrained models such as BERT [3] and GPTs [14] outperform other NLP methods and are applied for multiple tasks, including decision-making. [4] assessed ChatGPT responses on hazard analysis and suggested LLMs may be useful to assist human experts in this field. [15] used a pre-trained BERT and fine-tuned it for medical disease prediction. Such applications demonstrate the potential of LLMs for various decision-making tasks across different fields. However, applying them in specific domains requires delicate fine-tuning, as they were pre-trained on large common-sense corpora and consists of millions of parameters and their performance can be hindered when domain-specific data is insufficient. Additionally, responses from LLMs can be unreliable. Since their answers are generated by black-box processes, changes in prompts can lead to different outputs. In practice, a human expert is required to supervise the use of LLMs to ensure the correctness.

This paper proposes a transformer-based method which targets road maintenance decision-making processes and provides traceable and reliable decisions with step-by-step thinking. The decisions consist of four parts: the judgement of asset types, defect types, job types, and priority. The key difference between our approach and popular pre-trained LLMs such as BERT and GPTs is the attention mechanism (a method that enables the model to focus on the most important part of the input data). While BERT and GPTs primarily use self-attention mechanisms to focus on single source inputs only, our approach relies on a cross-attention mechanism, which enables connections between two different sources, to align road maintenance enquiry text (the description of road conditions requiring a maintenance activity) and maintenance knowledge. The contributions of this approach are summarized as follows:

1. Road maintenance knowledge is structured in logical chains, which imitate expert decision-making think path and provide visible and traceable steps for decision-making processes. The use of logical chains enables stepwise thinking within the model through the cross-attention mechanism.

2. The model is trained with real-world road maintenance records and achieves automatic maintenance priority rankings, together with asset, defect, and job information alignment.

3. This model, while using fewer parameters, outperforms pre-trained and fine-tuned BERT models, demonstrating the capacity of decision-making in road maintenance domain.

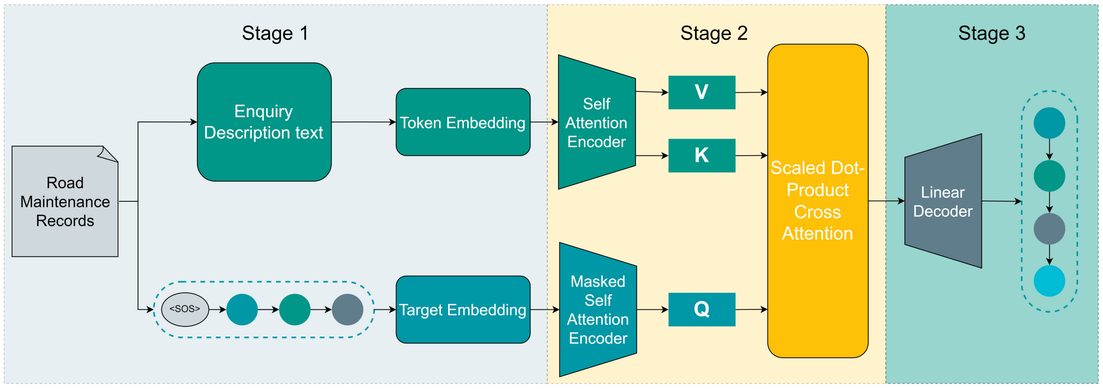

Figure 1: The architecture of RTransformer model.

2 Method

Figure 1 provides a high-level illustration of the architecture (named RTransformer standing for the Road-Transformer), consisting of three stages: (1) the processing of inputs, including texts and logical chains, before the attention mechanism, which involves formatting the logical chains and mapping inputs into a continuous vector space as embeddings. Both the enquiry descriptions and the logical chains are extracted from real-world road maintenance records. The logical chain is then arranged in a specific order, with a special placeholder ¡SOS¿ (Start of Sequence), inserted at the beginning. The discussion of the logical chain is covered in subsection 2.1; (2) the attention mechanism, including self-attention, where input information is integrated and prepared for the next layer, and cross-attention, where the logical chains can focus on the text information. A mask is added in logical chain processing to prevent data leakage. The V stands for Value, K stands for Key and Q stands for Query. The attention mechanism is discussed in detail in subsection 2.2; (3) the decoding and prediction of the next position in the chain, which is achieved using a linear decoder and a softmax layer to obtain the probability distribution of positions.

2.1 Road Maintenance Decision-Making Logical Chains

Road maintenance decision-making by human experts involves the intermediate judgement of maintenance-related factors and the final decision made on top of these factors. With their experience, experts can classify key factors before making maintenance decisions. Experts can intuitively identify the asset for which the maintenance enquiry is reported on, the defect which has occurred on the asset, and the job required to fix the defect. Given all this information, final decisions are based on the priority of maintenance tasks, which reflects their level of urgency and significance. Some examples of road maintenance decision-making key factors are listed in Table 2.

NLP methods are available to extract key factors from natural language. It is common practice to treat NLP-based decision-making problems as classification tasks, directly predicting final decisions such as activity priority. Such methods typically involve locating key words for decision-making, such as Named Entity Recognition (NER). However, road maintenance enquiries can be vague and written in layman's language, as it describes road conditions based not only on inspections but also on feedback from customers. The following example demonstrates

Asset Type	Defect Type	Job Type	Priority
Single 2-Lane Carriageway	Pavement Pothole	Incident Response	2 hours
Gully	Drainage Defect	Incident Repair	24 hours
Safety Barrier (Steel)	Miscellaneous Accident damage	Fences, Walls etc. Repair	2 months
Lightning Column	Lighting Column Defect	Lightning Repair	7 days
Post (Signs)	Road Markings Wear	Traffic Sign/Road Marks Repair	28 days
Catch pit	Drainage Flooding	Drainage, Service Ducts Repair	Fix now
Dual 1-Lane Pavement	Debris in traffic lane	Sweeping, Cleaning	14 days
......	......	......	......

Table 2: Examples of key factors of road maintenance.

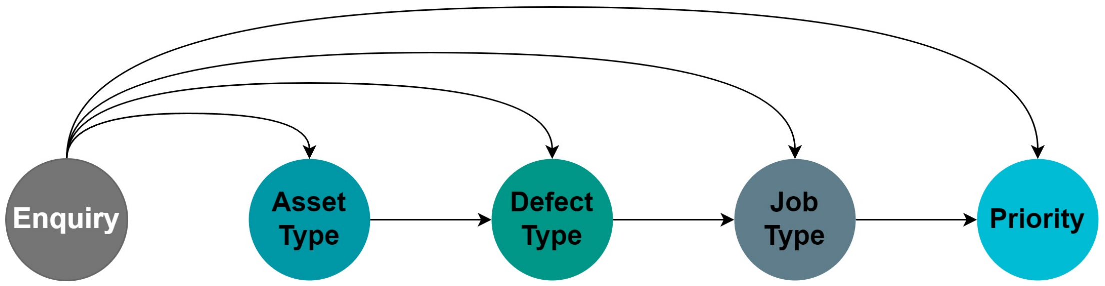

Figure 2: The logical chain of road maintenance.

part of an enquiry raised by a customer:

"Same problem .... Very loud clattering noise every time traffic goes over it. Sound like something metal which is loose ...... The noise every time a vehicle goes over it, is immense. I wouldn't be surprised also now that it could possible cause a serious accident......"

This enquiry describes a rattling manhole issue; however, no terms related to drainage are mentioned, and misleading words like "serious" and "accident" are used. In such cases, NLP methods that make direct decisions may overlook the complexity inherent in assessing multiple factors, leading to unexplainable and inaccurate outcomes.

To address this problem, we proposed a logical chain that manages knowledge in a structured format and mirrors the intuitive decision-making process of experts. An illustration is shown in Figure 2. The logical chain for road maintenance is designed in the order of Asset Type, Defect Type, Job Type and Priority, aligning with real-world practices. From the enquiry, the type of asset is first confirmed, and then the type of defect is reasoned based on both enquiry content and asset information. By repeating this process, the logical chain enables each subsequent prediction to benefit from the additional information provided by all previous predictions until the final decision of priority is made. Taking the previous customer raised enquiry as an example, "Manhole" is first matched as the asset type, then "Drainage Defect" is reasoned as the defect type with additional asset information, followed by the job type as "Drainage, Service Ducts" follows. Finally, the priority is decided as "7 days", considering all previously extracted information.

2.2 Attention Mechanism

The attention mechanism used in this work is Scaled Dot-Product Attention[21] as used in the transformer model, which can be expressed as Equation 1. Q (for query), K (for key) and V (for value) are vectors that represent inputs in continuous space. The dimension of vectors is denoted as d, and $\sqrt{d}$ is used as a scaling factor.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{1}$$

The self-attention mechanism is wrapped in encoders to separately process inputs from the enquiry text source and the knowledge source. In the self-attention mechanism, Q, K and V are derived from the same source and contribute to the vector v that assembles all token features. In the self-attention encoder on the text side, the output v_{text} is subsequently used as K and V in the following cross-attention mechanism. On the knowledge side, additional masking is added to prevent data leakage. As the positions of the logical chain are the targets to predict, the current prediction depends on all known previous positions. The output $v_{knowledge}$ provides Q for the cross-attention mechanism. Hence, for cross attention mechanism, the original attention equation can be rewritten as Equation 2:

$$\text{Attention}(v_{knowledge}, v_{text}) = \text{softmax}\left(\frac{v_{knowledge}v_{text}^T}{\sqrt{d}}\right)v_{text} \tag{2}$$

Equation 2 is used in cross-attention for text-knowledge alignment, where i is the index of the position to predict and $i \in 1, 2, 3, 4$. The data flow from self-attention to cross-attention is illustrated in Figure 3 using a real-world example. The enquiry description text "Road Traffic Collision CAR HIT LAMP COL" is first tokenized into separate tokens, and the token embedding layer generates embeddings for each token, represented as $[x_1, x_2, ..., x_d]$. Through self-attention mechanism, v_{text} is generated based on all token embeddings. On the other hand, $< SOS >$ is concatenated at the beginning of the logical chain as a placeholder for cross-attention. Through the target embedding layer, positions and $< SOS >$ are embedded as $[y_1, y_2, ..., y_d]$. Self-attention mechanism also processes the embeddings of positions less than i (because of masking) and outputs $v^i_{knowledge}$. Cross-attention mechanism, v_{text} attends to $v^i_{knowledge}$ of each position for a step-by-step prediction. For example, to predict Job Type position, masked self-attention accumulates data from $< SOS >$ embedding, "Lighting Column" embedding at Asset Type position, and "Miscellaneous Accident damage" embedding at Defect Type position, and outputs $v^3_{knowledge}$ as the result of self-attention. Data from "Incident Response" and "30 Minutes" embeddings at position 3 and 4 are not considered. Together with v_{text} from self-attention for texts, the attention is computed by Equation 2 and then decoded by the linear decoder to predict the exact content at Job Type position.

2.3 Training and Reasoning Processes

In this section, we introduce the training and reasoning processes employed by our RTransformer. Both processes follow the path of stepwise prediction, but the strategy of training and reasoning is different. During training, a teacher forcing strategy is used to guide the model learning process with ground truth data. In the reasoning process, ground truth is not available, and the subsequent predictions depend on the previous predictions. These different strategies allow the model to be efficiently trained and to make predictions independently.

2.3.1 Training Process

The teacher forcing strategy is shown in Figure 4. The attention mechanism is employed among ground truth rather than predictions. During the training process, the Enquiry text, $< SOS >$ placeholder and ground truth are the inputs to RTransformer model. To implement teacher

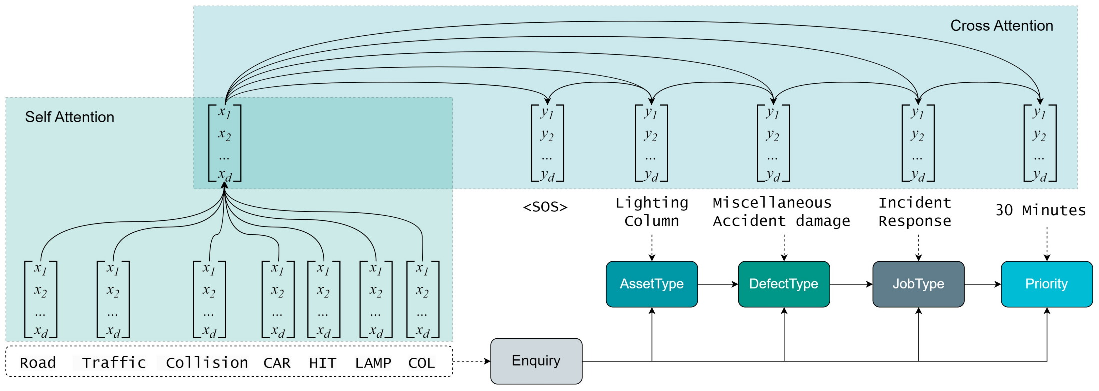

Figure 3: Data flow in the combined attention mechanism.

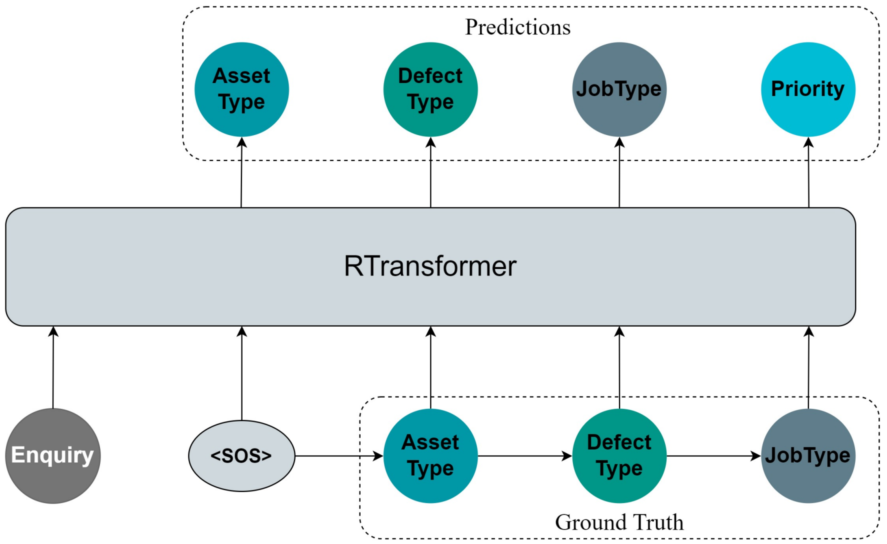

Figure 4: Teacher forcing in RTransformer training process.

forcing, each position of the logical chain is shifted by one. The last position is dropped, and the beginning of the chain is held by $< SOS >$. The first prediction is made from Enquiry and $< SOS >$. Instead of using the predicted position, the next prediction depends on the ground truth of the previous position, e.g. Defect Type prediction is made from Enquiry, $< SOS >$ and ground truth Asset Type. In this way, the model always learns from the correct knowledge.

2.3.2 Reasoning Process

The reasoning process can be described in pseudocode as shown in Figure 5. The major difference is that the inputs to the RTransformer on the knowledge side are predictions rather than ground truth. The RTransformer model and $< SOS >$ are pre-trained and their parameters are frozen during reasoning. The data flow on the Enquiry text side remains the same as in the training process. On the knowledge side, the input is initialized from the placeholder $< SOS >$

```
Algorithm 1 Reasoning Process
1:  Input: Enquiry text
2:  Retrieve the pre-trained model RTransformer, and the pre-trained Placeholder <SOS>
3:  Tokenize Enquiry, applying truncation or padding to achieve the predefined length
4:  Initialize Target using Placeholder <SOS>
5:  Initialize Position array
6:  for i from 0 to 3 do
7:      Calculate Logits using RTransformer(Enquiry, Target)
8:      Predict Position_i class from softmax(Logits)
9:      Concatenate the predicted Position_i to the end of Target to form the new Target
10: end for
11: Return Position
```

Figure 5: Pseudocode of Decision Reasoning Process.

and dynamically updated with the predictions in the reasoning loop. The reasoning loop stops after four predictions, concluding the decision-making process. The predicted logical chain is the final output of RTransformer.

3 Experiments

This section describes the RTransformer training process in detail, together with comparison to BERT variants as baseline models. This model is trained on a virtual machine with 1 NVIDIA Tesla V100 GPU.

3.1 Training data

The RTransformer and the baseline models are both trained on real-world road maintenance records from National Highways England [9]. The original data was cleaned and balanced across classes for a better quality. The total amount of records used for training is 19,297, and each record contains enquiry description texts, the Asset Type, the Defect Type, the Job Type and the Priority. A logical chain is generated for each record. For enquiry texts, the sequence length is limited to 100 tokens, as over 99% sequences are shorter than this threshold. A special token ¡PAD¿ is used for padding. The shared vocabulary in the token embedding layer consists of 9,231 tokens. For the target positions to predict, there are 125 classes in total across all four positions.

3.2 Baseline Model

We used BERT as our baseline model to compare the performance on the same training dataset. As BERT uses the encoder only architecture and focuses on self-attention, we simplified the decision-making problem as a last step classification task. In this way, the original step-by-step decision-making of 125 classes is turned into one final priority classification of 30 classes. With fewer classes, it is easier for pre-trained BERT to differentiate between classes. In our experiments, BertForSequenceClassification model from Hugging Face library[22] is used, and the pre-trained weights are from different versions of BERT-based models with different sizes.

3.3 Optimizer and Hyperparameters

We used the AdamW optimizer [12] instead of the Adam optimizer [11] that is used in the original transformer. The change was made to address the overfitting issue caused by insufficient training data. The hyperparameters are set as follows: $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, weight_decay= 10^{-4},learning_rate = 10^{-4}. A dropout rate of 0.1 is also applied in the model.

In this work, we also employ Multi-Head Attention to jointly attend to information in different aspects at different positions. A test of different head numbers and layers is also conducted to find an optimal model structure.

4 Results

This section presents the results of experiments of testing RTransformer in different variations and the performance of baseline models.

4.1 Model Variations

The experiments are conducted based on an early stop strategy. The whole dataset is divided into two subsets, with 70% of the data allocated for training and 30% reserved for validation. The patience parameter in the early stopping strategy is set to 5, allowing a wait of five epochs for any improvement in validation loss.

Variations of RTransformer have been tested in different aspects, including the change of layer numbers, batch sizes, number of heads and the dimension of models. We denote the attention layer number in encoders and cross-attention block as layer, the dimension of $v_{knowledge}$ and v_{text} before Multi-Head Attention application as dmodel, the dimension of the feed forward layer in the linear decoder as dff. In Table 3, the params column records the amount of all trainable parameters in RTransformer. The epoch corresponds to the last instance before the validation loss begins to increase, indicating potential overfitting. And the accuracies of the four positions are also listed in the table. All accuracy metrics are obtained in the validation dataset at the epoch corresponding to the minimum validation loss.

	layer	batch	d_{model}	d_{ff}	head	params $\times 10^6$	epoch	AssetType accuracy	DefectType accuracy	JobType accuracy	Priority accuracy
base	6	256	512	2048	8	51.4	10	0.453	0.674	0.803	0.610
(A)	3					29.4	10	0.465	0.697	0.810	0.626
	2					22.0	11	0.483	0.701	0.809	0.632
	1					14.6	11	0.473	0.673	0.790	0.610
(B)	2				1	22.0	12	0.459	0.672	0.785	0.610
	2				4	22.0	12	0.482	0.697	0.806	0.631
	2				16	22.0	12	**0.511**	**0.727**	**0.827**	**0.650**
	2				32	22.0	12	0.484	0.701	0.810	0.626
(C)	2		256			9.6	20	0.499	0.718	0.817	0.643
	2		768		16	37.5	8	0.476	0.699	0.805	0.627
(D)	2	64			16	22.0	7	0.478	0.683	0.805	0.616
	2	128			16	22.0	8	0.482	0.705	0.807	0.625

Table 3: Variations on the RTransformer architecture and experiment results.

The base model architecture is configured to align with the original transformer base model settings. Due to the significantly smaller size of the training dataset compared to the original transformer, it is essential to adjust the number of layers and heads to identify the optimal configuration. A balance is also required between accuracy performance and the number of parameters, which directly impacts computational cost.

The result summary of model variations is listed in Table 3. Group (A) includes the test on how different numbers of layers influence the result. The best result is gained when two layers are used, with 22.0 million parameters. The model with more layers does not achieve better performance, which may result from the misalignment between the training data volume and the parameter volume. When the number of layers is reduced to one, the model does not fully utilize its potential to learn from the data. Group (B) explores the influence of heads. As heads allow the model to attend to different aspects of the data, the model with 16 heads achieves the best result, while using 32 heads leads to earlier overfitting. The test in group (C) examines the variations in the model dimensions. A dimension of 256 leads to the disqualification of the model, while an increase in dimensions results in early overfitting. The test in group (D) focuses on the batch size, which influences the stability of the training. A bigger batch size improves the performance with fewer learning steps in each epoch. However, limited by the memory of the GPU, the biggest available batch size is 256.

The results presented in Table 3 demonstrate a trend of overfitting. Although the vanilla Transformer achieves the best performance with 213 million parameters [21] in their experiments, the bigger the better does not apply to domain-specific applications. One of the primary reasons is the deficiency of training data both in terms of quantity and diversity, which easily triggers the early stop strategy to prevent overfitting.

4.2 Comparison to Baseline Models

BERT-based pretrained models are used as baseline models to compare performance on priority prediction. As these models are pretrained on lots of various books and Wikipedia, fine-tuning is required to adapt them for the road maintenance domain. This fine-tuning dataset is the same dataset used in RTransformer training.

BERT variants of different sizes are tested on the same dataset. The results are listed in Table 4. BERT$_{\text{base}}$ is the original BERT model from [3], the DistilBERT model leverages knowledge distillation and reduces the model size to 65.8 million [18]. To compare performance across various model sizes, BERT$_{\text{medium}}$ and BERT$_{\text{small}}$ [20] are also tested.

Due to the limitation of GPU memory, the training batch size is unified as 128.

	layer	batch	d_{model}	d_{ff}	head	params $\times 10^6$	epoch	Priority accuracy
ours	2	128	512	2048	16	22.0	8	**0.625**
BERT$_{\text{base}}$	12	128	768	3072	12	110	4	0.580
DistilBERT	6	128	768	3072	12	65.8	5	0.476
BERT$_{\text{medium}}$	8	128	512	2048	8	41.7	4	0.582
BERT$_{\text{small}}$	4	128	512	2048	8	29.1	5	0.583

Table 4: Performance comparisons among BERT variants and KGTT

From Table 4, we can see that our model achieves the best performance with the smallest model size. Additionally, the accuracy of our model is gained on the base of 125 classes of all positions, while other BERT variants are tested with 30 classes from the priority position only. Apart from DistilBERT, which utilizes a different mechanism, the size reduction of BERT models has little impact on accuracy results. This may be attributed to the quality and quantity of fine-tuning data.

5 Conclusions

This paper leverages a transformer to conduct decision-making in road maintenance. The cross-attention mechanism is applied to align text and domain-specific knowledge. By integrating the cross-attention mechanism with the logical chain design, this approach achieves a stepwise decision-making process. The results of experiments on real-world data demonstrate that RTransformer outperforms BERT models that utilize self-attention alone.

In domain specific tasks such as road maintenance decision-making, the limited availability of data restricts the performance of models, often leading to overfitting during training and fine-tuning. In this context, a restrained model architecture is preferable. Although pretrained LLMs are considered suitable solutions for knowledge management, they may not be optimal choices in domain-specific tasks, especially when considering data availability and computational costs.

The method proposed in this paper demonstrates the potential of utilizing the Transformer architecture for decision-making tasks in specific domains. It provides a feasible solution to guide the learning process of the model by designing logical chains to allow stepwise thinking.

Despite the promising results achieved in this study, several research limitations remain. Although the stepwise thinking introduced by the logical chain improves the performance of our model, it relies on semantic relations among elements and overlooks the structural features of knowledge. Additionally, the format of logical chains lacks compatibility. Future work will aim to organize knowledge in a more structured manner and incorporate structural features into the decision-making process.

6 Acknowledgments

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/V056441/1], the first author is sponsored by the China Scholarship Council and Cambridge Trust for PhD study at the University of Cambridge. The authors of this paper would like to thank Lavindra de Silva, Alix Marie d'Avigneau and Haritha Jayasinghe for their comments and inspiration.

References

[1] Asphalt Industry Alliance. Annual Local Authority Road Maintenance Survey Report 2023.

[2] Breno Hvala de Figueiredo, Marcos dos Santos, Luiz Paulo Lopes Fávero, Miguel Ângelo Lellis Moreira, and Igor Pinheiro de Araújo Costa. Analysis of maintenance activities in Urban Pavement Management Systems based on Decision Tree Algorithm. *Procedia Computer Science*, 214:712–719, 2022.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

[4] Simon Diemert and Jens H. Weber. Can Large Language Models Assist in Hazard Analysis? In Jérémie Guiochet, Stefano Tonetta, Erwin Schoitsch, Matthieu Roy, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security. SAFECOMP 2023 Workshops*, pages 410–422, Cham, 2023. Springer Nature Switzerland.

[5] Marco D'Orazio, Elisa Di Giuseppe, and Gabriele Bernardini. Automatic detection of maintenance requests: Comparison of Human Manual Annotation and Sentiment Analysis techniques. *Automation in Construction*, 134:104068, February 2022.

[6] Marco D'Orazio, Gabriele Bernardini, and Elisa Di Giuseppe. Automated Priority Assignment of Building Maintenance Tasks Using Natural Language Processing and Machine Learning. *Journal of Architectural Engineering*, 29(3):04023027, 2023.

[7] Department for Transport. Transport Statistics Great Britain, 2022.

[8] Georgios M. Hadjidemetriou, Johannes Masino, Symeon E. Christodoulou, Frank Gauterin, and Ioannis Brilakis. Comprehensive Decision Support System for Managing Asphalt Pavements. *Journal of Transportation Engineering, Part B: Pavements*, 146(3):06020001, 2020.

[9] National Highways. National Highways - National Highways, September 2023. Archive Location: Worldwide Publisher: National Highways.

[10] Highways England. Highways England Strategic Road Network Initial Report - Overview, 2017.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. arXiv:1412.6980 [cs].

[12] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization, November 2017.

[13] Danial Moazami, Hamid Behbahani, and Ratnasamy Muniandy. Pavement rehabilitation and maintenance prioritization of urban roads using fuzzy logic. *Expert Systems with Applications*, 38(10):12869–12879, September 2011.

[14] OpenAI. ChatGPT.

[15] Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-BERT: Pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *npj Digital Medicine*, 4(1):1–13, 2021.

[16] Richard Robinson, Uno Danielson, and Martin Snaith. *Road Maintenance Management*. Macmillan Education UK, London, 1998.

[17] Hassim Salihudin, K.T. Teh, Ratnasamy Muniandy, Hatem Omar, and A. Hassan. A Prototype Expert System for the Selection of Road Construction Materials. *The Journal of Engineering Research [TJER]*, 4:1, December 2007.

[18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, October 2019.

[19] K P Tripathi. A Review on Knowledge-based Expert System: Concept and Architecture. *Artificial Intelligence Techniques*, 2011.

[20] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-Read Students Learn Better: On the Importance of Pre-training Compact Models.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[22] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. HuggingFace's Transformers: State-of-the-art Natural Language Processing, July 2020. arXiv:1910.03771 [cs].