Tag infilling for Vietnamese administrative forms: Leveraging synthetic datasets and domain-specific modelling

Vinh Nguyen[1, 2], Tung Le[1, 2], Huy Tien Nguyen[1, 2*]

[1]*Faculty of Information Technology,University of Science Vietnam National University Ho Chi Minh City, 227 Nguyen Van Cu Street, Cho Quan Ward, Ho Chi Minh City, Vietnam*

[2]*Vietnam National University Ho Chi Minh City, Vo Truong Toan Street, Linh Xuan Ward, Ho Chi Minh City, Vietnam*

Received 11 November 2025; revised 31 August 2026; accepted 26 September 2026

Abstract:

Automatic identification of administrative tags is vital for automating administrative form processing. Despite its importance, this task remains underexplored in low-resource languages and domain-specific settings, especially in the context of Vietnamese government documents. In this article, we introduce a comprehensive and scalable pipeline for assisting constrained administrative tag infilling in Vietnamese administrative forms, comprising synthetic dataset construction, domain-adapted model training, and robust evaluation strategies. This forms a complete pipeline that predicts the appropriate administrative tag from a predefined tag set for a variety of forms generated from authentic Vietnamese administrative documents. Using a large-scale dataset of 50,000 synthetic Vietnamese administrative forms annotated with 425 distinct tags and structured into five levels of complexity, our experiments show that domain-adapted Transformer-based models consistently outperform general-purpose zero-shot large language models on seen data and generalise well to unseen inputs. This highlights the potential of our approach to achieve high accuracy with substantially smaller models, offering a more practical and efficient solution for processing Vietnamese administrative forms.

Keywords: **administrative documents, large language model, masked language model, tag infilling, text infilling.**

Classification number: **1.2**

[*]*Corresponding author: Email: ntienhuy@fit.hcmus.edu.vn*

1. Introduction

In recent years, as large language models (LLMs) continue to evolve, natural language processing (NLP) tasks are being applied to many aspects of daily life, especially in the digital transformation efforts of governments and organisations. When converting paper documents to digital formats, automated tools are essential for efficient document processing. One critical NLP technique for this purpose is text infilling, which fills in missing information to improve both processing speed and accuracy. However, applying this technology to low-resource languages like Vietnamese presents many challenges, such as semantic complexity and limited data availability. In this study, our primary goal was to apply text infilling by predicting administrative tags to Vietnamese documents, particularly within the context of national administrative procedures.

The text infilling task aims to replace blanks in forms with one or more character sequences that appropriately fit the sentence or document context. In the administrative setting considered in this study, the missing content is represented by predefined administrative tags, making this a constrained form of text infilling. For example, when Vietnamese citizens engage with public administrative services, they are often required to fill out numerous forms across different procedures, each demanding the same repetitive personal information such as full name, date of birth, gender, ID number, and address. This manual and redundant process not only consumes time but also increases the likelihood of input errors. Administrative tag infilling offers a practical solution by automatically completing these frequently requested fields. In general-purpose systems, these fields are represented using placeholder tags like <FullName> or <DateOfBirth>, which can later be replaced with actual user data retrieved from existing government databases. Accurate administrative tag infilling can therefore significantly streamline the form-filling experience, reduce user effort, and improve the overall efficiency of digital public services. In practice, the overall text infilling task bears a strong resemblance to sentence or passage completion, a problem that has been widely studied in both linguistics and natural language processing. Traditional approaches often rely on symbolic parsers that utilize contextual cues and linguistic features such as part-of-speech (POS) tags or dependency structures to infer missing content [1]. More advanced methods, such as the work by P. Mirowski, et al. (2015) [2], leverage recurrent neural networks

(RNNs) combined with syntactic representations to model dependencies for sentence completion. With the advent of attention mechanisms and Transformer-based architectures, recent models have achieved substantial improvements in infilling accuracy by better capturing long-range dependencies and contextual semantics. These advances make it increasingly feasible to apply tag infilling in real-world applications, such as Vietnamese administrative form processing, where high accuracy and contextual relevance are essential for reducing manual effort and ensuring reliable automation.

Recent advances with Transformer-based models have greatly improved the ability to fill in missing information in text. However, applying these techniques to Vietnamese administrative documents is still a relatively new area of research. These official forms, which are used in various citizen and government interactions, cover a wide range of topics such as housing, civil status, and personal information registration. Although there is a variety of forms and tag types, the total number of available documents is not large enough to build a large-scale dataset for training specific language models for this area. Furthermore, many forms include tags that have similar meanings or refer to related concepts (e.g., the applicant's name, father's name, and mother's name), which makes it difficult for a language model to accurately tell them apart. This creates additional challenges for administrative tag infilling systems, especially when the model needs to understand fine-grained differences in meaning based on context.

The complexity of this task is further compounded by a well-known limitation of LLMs: their tendency to prioritise fluency and exhibit high confidence in predictions, even when uncertain [3-5]. This overconfidence can become problematic in text infilling tasks involving Vietnamese administrative forms, where selecting the wrong tag can lead to misleading or incorrect results. Additionally, some tags are rare and appear only in specific documents, further complicating the model's ability to accurately infer or complete missing information.

To address these challenges, we proposed a comprehensive pipeline that includes: (1) curating and preprocessing authentic Vietnamese administrative forms collected from the National Public Service Portal; (2) generating a synthetic dataset to supplement the limited real-world data; (3) training a domain-adapted

Transformer-based language model on this data; and (4) evaluating the model across multiple benchmarks. Our experimental results showed that the proposed model significantly outperforms competitive baselines, including ChatGPT and Gemini under zero-shot prompting, in accuracy when filling in administrative forms. The contributions of our study can be summarised as follows. First, we construct a real-world dataset that reflects the attributes and formats of documents from Vietnam's National Public Service Portal. Second, we design a complete pipeline, from training to evaluation, to assess the performance of both language models and LLMs. Third, we demonstrate that our domain-adapted language model outperforms zero-shot prompting approaches using general-purpose LLMs such as ChatGPT and Gemini, particularly in flagging unseen tags, which are tags not present in the training set.

The structure of this article is organised as follows: Section 2 discusses text infilling and administrative tag infilling, related research, and the processing of administrative data. Section 3 details our methodology, including text generation, model training, and evaluation methods. Section 4 presents our results and analysis. Finally, Section 5 summarises our conclusions and suggests directions for future research.

2. Literature review

Administrative forms are widely used in official procedures, yet remain understudied in natural language processing, primarily due to their structural complexity and the limited data availability in low-resource languages. Recent advances in LLMs have enabled the generation of synthetic text that can be leveraged for training downstream text infilling models. However, the use of LLMs for data generation and of language models for tag infilling in administrative forms, especially in Vietnamese, remains largely unexplored.

2.1. Administrative data

Administrative forms are standardised documents issued by agencies, organisations, and governments to collect or standardise information for official procedures and processes. Due to their complexity and frequent use, machine learning has been applied to enhance the efficiency of processing such forms. A. Hamdi, et al. (2018) [6] proposed a document stream segmentation model based on a dataset of administrative texts. S. Biswas, et al. (2021) [7] introduced a graph-

based generative model for generating administrative invoice data. X. Ling, et al. (2020) [8] developed a framework integrating optical character recognition (OCR) and natural language processing (NLP) methods to process administrative documents. These studies indicate that machine learning-based administrative document understanding is becoming increasingly prevalent. However, these studies primarily focus on tasks such as segmentation, data extraction, or document generation, and do not address the problem of completing missing fields through text infilling. Our work filled this gap by introducing a pipeline for administrative tag infilling in Vietnamese administrative forms.

2.2. Data generation

Since LLMs have emerged as AI tools applicable across various domains, numerous studies have explored their use in text data generation tasks. However, there remains a performance gap between closed-source and open-source LLMs. Z.A. Nazi, et al. (2025) [9] showed that while these models can perform comparably on certain specific tasks, closed-source models such as ChatGPT generally achieve better overall performance. Moreover, H. Gurdil, et al. (2024) [10] showed that ChatGPT can generate high-quality data in educational contexts. H. Tari, et al. (2024) [11] highlighted the potential of AI-generated data, although its downstream performance can vary compared to human-authored datasets. S.A. Lehr, et al. (2024) [12] demonstrated that while ChatGPT may not produce novel scientific insights, it effectively replicates existing patterns. Extending this line of work, we utilised LLMs both to generate synthetic Vietnamese administrative form data and to serve as baseline systems for benchmarking. This dual use allowed us to evaluate the benefits of domain-adapted Transformer-based models over general-purpose LLMs in this specialised setting.

2.3. Text infilling

Earlier, text infilling originated from masked sentence prediction, which became a widely used objective in NLP tasks. K. Song, et al. (2019) [13] proposed the MASS model, which adopts a masked sequence-to-sequence training strategy, enabling the model to generate contextually accurate spans that align well with the requirements of automatic document completion. M. Lewis, et al. (2020) [14] proposed the BART model, trained using multiple noising schemes, including an infilling objective, demonstrating that language models can effectively

understand incomplete input and generate coherent output through infilling pre-training. W. Zhu, et al. (2019) [15] proposed a segment-aware position encoding mechanism combined with a self-attention model to fill in missing content. C. Donahue, et al. (2020) [16] introduced a language model-based infilling approach that enables effective information completion across diverse data types. Additionally, D. Li, et al. (2022) [17] developed the A-TIP model, which employs an adaptive discriminator to guide the infilling process based on text attributes. These studies have contributed to improving text infilling quality, evolving the task from sequence-to-sequence models to more advanced Transformer-based architectures capable of controlled and attribute-aware text generation. While prior studies have advanced text infilling for general NLP tasks, they largely overlook low-resource languages and administrative domains. Our work instead focused on Vietnamese administrative forms, a domain and language combination that remains underexplored. Inspired by the infilling paradigm, our task was formulated as selecting the correct tag from a fixed, predefined set to fill a masked slot, rather than generating free-form text. This formulation better reflects the constrained and standardised nature of administrative documents, where allowable content is limited and must remain precise.

3. Methodology

3.1. Data collection and generation

Due to the variability in the format and content of administrative forms across different sources, we ensured accuracy by collecting them exclusively from Vietnam's National Public Service Portal. For practical purposes, we prioritised the most commonly used forms, as ranked on the portal. After collecting each form, we extracted all associated tags. This procedure was applied to 50 distinct forms, resulting in a total of 425 unique tags. To further enrich the dataset, we used the ChatGPT API to generate additional documents.

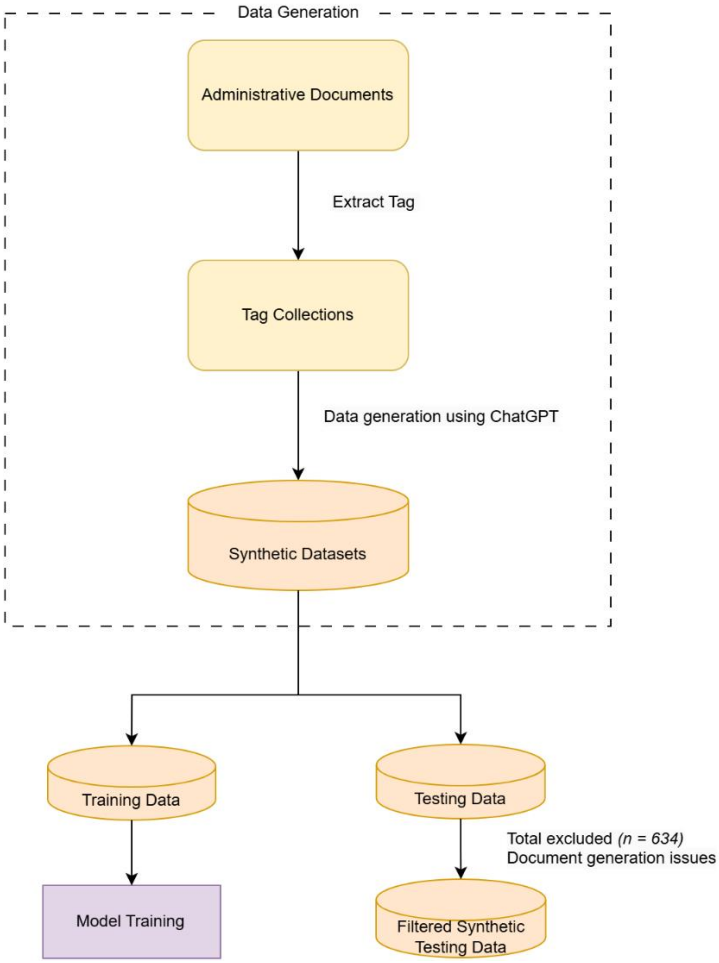


Fig. 1. Data generation and modelling pipeline.

As described in Fig. 1, we generated synthetic data for each type of administrative form individually. Specifically, for every new form generated, we prompted ChatGPT based on a predefined tag collection. An example prompt used to generate a form:

"You are an administrative officer specialising in processing government administrative forms in Vietnam and forms from various ministries and departments. Given the following list of tags, follow these instructions:

(1) Create a Vietnamese administrative form using the tags (Note: You are not allowed to change tag content and you should preserve tag format as provided). The form should consist of multiple sentences that clearly express the content of the document, not just descriptive phrases of the tags.

(2) Base the content on existing administrative forms in Vietnam.

(3) Present the forms as plain text paragraph, without requiring national symbols or official headings or new line. Focus solely on the textual content of the form.

(4) Ensure diversity in the content of the text, allowing the order of tags to be rearranged. You may use synonyms or multiple ways of phrasing expressions.

Tag list: Tag_1, Tag_2, ..., Tag_N" where Tag_N is a candidate tag.

Each iteration produced 1,000 synthetic forms, amounting to a total of 50,000 synthetic forms derived from 50 original forms. These documents were then divided into five datasets, as described in Table 1, referred to as Infill Levels (IL) 1 through 4 and a final Complex level. Each level represented an increase in dataset complexity, reflected not only by the number of tags but also by the semantic similarity among the tags, which made the infilling task progressively more challenging. Specifically, the Infill Complex dataset was generated from a set of tags randomly selected from 50 sets of tags. From each set, one or two unique tags were randomly chosen to avoid excessive information loss, resulting in a total of 77 unique tags for generating the Infill Complex document. These tags were then split into an approximate 80/20 ratio, comprising 61 and 16 tags, respectively. The 61 tags were designated as exclusive tags for generating the Infill Complex dataset and are not used in the generation of documents for IL 4.

Table 1. Training, validation, and test sizes across infill levels.

Dataset	IL 1	IL 2	IL 3	IL 4	Infill complex
Number of forms	5	10	25	50	-
Train	3,000	6,000	15,000	30,000	-
Validate	1,000	2,000	5,000	10,000	-
Test	985	1,927	4,856	9,175	423

3.2. Model training

By structuring the data into five datasets with an increasing number of tags, we enabled a comparative analysis of the robustness of domain-adapted Transformer-based models against prompting large language models in filling in missing tags. For each dataset, we split the data into training and test sets, with a subset of the training data used for validation, as described in Table 1. In the test set, we manually cleaned the data by comparing words generated by ChatGPT with a Vietnamese dictionary [18] to identify incorrect words. We also removed

documents that contain erroneous tags due to ChatGPT's hallucinations. In total, 634 documents were removed from the testing data, accounting for 3.52% of the test set. This ensured that in IL 1, 2, 3, and 4, the number of tags remained consistent between the training and test sets. Specifically, IL 1 included 98 tags from five types of forms; IL 2 included 104 tags from 10 types of forms; IL 3 included 191 tags from 25 types of forms; and IL 4 included 359 tags from 50 types of forms.

However, in practice, Vietnamese administrative forms often contain unique pieces of information that appear in only one or a few specific forms. To assess the ability of both domain-adapted Transformer-based models and LLMs to recognise previously unseen information, we created Infill Complex, which contained tags that were not present in the training set. Specifically, we used the model trained on IL 4 to test on this dataset, meaning that unseen tags refer to tags outside of the Infill Level 4 training data. As a result, Infill Complex comprised two types of tags: 16 seen tags and 61 unseen tags. This design allowed us to evaluate whether the language models could correctly recognise the absence of known information or would instead infill incorrect tags in such cases.

Unlike standard language models, masked language models learn from bidirectional context (both preceding and succeeding words relative to the [MASK] position). The objective function is computed by summing the loss at each [MASK] token in the document. We used the Data Collator object from the HuggingFace library to randomly replace tokens with [MASK] at a rate of 15%. After preprocessing, the text was fed into the model for training, using the following objective function:

$$\mathcal{L}_{MLM} = -\sum_{t \in M} \log P(x_t \mid X_t; \theta)$$

where M is the set of masked positions; X_t is the input sequence with x_t replaced by a special [MASK] token; $P(x_t \mid X_t; \theta)$ is the probability of predicting the masked token given the surrounding context.

4. Experiments

In this section, we present our evaluation methods, experiments, and their results. After describing our evaluation procedure and the models' parameters, we first show the results on datasets containing only seen tags. Next, we present detailed results of the models for groups of tags with similar meanings to assess the models' ability to distinguish between these tags. Then, we discuss the model's confidence level through the probability scores it outputs for unseen tags and provide an overall evaluation on datasets containing both seen and unseen tags. Finally, we demonstrate that our model did not make overconfident predictions on unseen data.

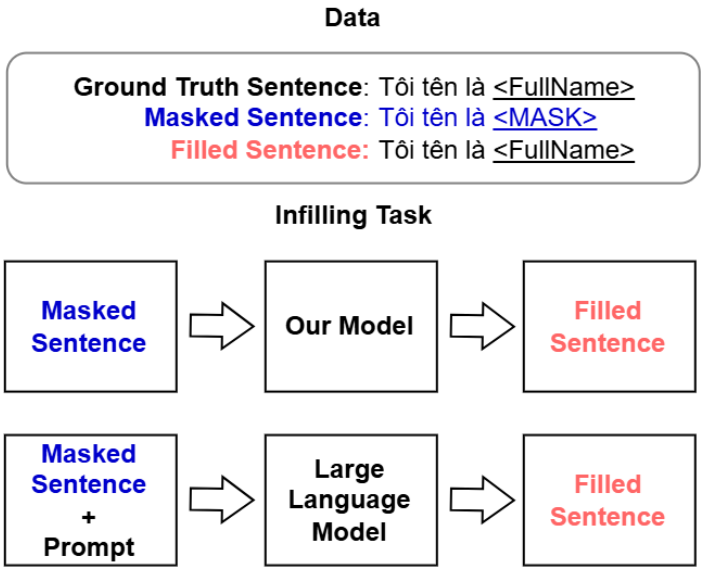


Fig. 2. Administrative tag prediction pipeline.

In our experiments, we evaluated our models and also compared their performance against two large language models: ChatGPT and Gemini. As illustrated in Fig. 2, we masked every tag in each sentence with <MASK> during evaluation. These masked sentences were then fed into our models to recover the missing information. For LLMs, we used a prompt that contained instructions, a list of candidate tags, and masked sentences for completion.

Both groups of models were evaluated on four test sets (from IL 1 to IL 4) and the entire Infill Complex using this scheme. However, since Infill Complex contains unseen tags, its evaluation differed from that of seen tags. Specifically, we relied on the confidence score, which is determined by the highest probability of the token output after the softmax function. For our models, if the probability

is less than 0.5 and the ground truth tag is an unseen tag, it was considered a correct prediction; otherwise, it was incorrect. The process was expressed by the following formula:

$$f(\hat{\mathbf{Y}}, y) = \begin{cases} \langle \text{ Unknown } \rangle & \text{if } y \in \mathbf{U} \text{ and } P(\hat{y}) < 0.5, \\ \hat{y} & \text{otherwise.} \end{cases} \quad (1)$$

where $\hat{\mathbf{Y}}$ is the set of predicted token probabilities; y is the ground truth label for the current instance; $\mathbf{U}$ is the set of unseen tags; $\hat{y} = \hat{Y}_{\arg\max(\hat{Y})}$ is the predicted token with the highest probability; and $P(\hat{y})$ is the probability of the predicted token after the softmax function.

For large language models, in addition to filling in the seen tags, we introduced a special tag '<Unknown>' in the prompt for cases where the model determines that a tag is unseen. An example prompt used to fill the form:

"In this task, you should replace <mask> with word that makes sense in the context (in format <word>). You have an extra list of words to include in your selection, you are not required to select a tag from the provided list. If the information was not given, fill <Unknown>.

Tag list: Tag_1, Tag_2, ..., Tag_N

Paragraph content: Content" where Content is the input paragraph content.

After having a model prediction, to determine whether that prediction is correct, we considered two scenarios based on the nature of the ground truth tag. In the first case, if the ground truth tag y belongs to the set of unseen tags U, the prediction is considered correct if the model outputs the special token <Unknown>. For our models, this corresponds to the evaluation function:

$$f(\hat{\mathbf{Y}}, y) = \langle \text{ Unknown } \rangle$$

As previously defined, for LLMs, the output string had to be explicitly <Unknown> in this case. In the second case, if the ground truth tag $y \notin U$, the prediction is considered correct only if the predicted tag $\hat{y}$ exactly matches the ground truth tag. The evaluation criteria were formalised separately for each model type as follows.

Our models:

$$\text{Correct}_(y, \hat{\mathbf{Y}}) = \begin{cases} 1 & \text{if } y \in \mathbf{U} \text{ and } f(\hat{\mathbf{Y}}, y) =< \text{ Unknown } >, \\ 1 & \text{if } y \notin \mathbf{U} \text{ and } \hat{y} = y, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

LLM:

$$\text{Correct}_(y, \hat{y}) = \begin{cases} 1 & \text{if } y \in \mathbf{U} \text{ and } \hat{y} =< \text{ Unknown } >, \\ 1 & \text{if } y \notin \mathbf{U} \text{ and } \hat{y} = y, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

Based on the above evaluation functions, we reported model performance in two categories: accuracy on seen tags, unseen tags, and Macro F1 on seen tags. IL 1-4 contained only seen tags and were evaluated accordingly. Infill Complex included both seen and unseen tags, allowing for a more comprehensive evaluation. The accuracy metrics were defined as follows:

$$\text{Accuracy Seen } = \frac{C_{\text{Seen}}}{N_{\text{Seen}}} \quad (4)$$

$$\text{Accuracy Unseen } = \frac{C_{\text{Unseen}}}{N_{\text{Unseen}}} \quad (5)$$

where C_{Seen} , C_{Unseen} are the number of correctly predicted seen and unseen tags, respectively; N_{Seen} , N_{Unseen} are the total number of seen and unseen tags in the dataset.

The Macro F1 metric was defined as follows:

$$\text{Macro F1}_{\text{Seen Tags}} = \frac{1}{K_{\text{Seen}}} \sum_{k=1}^{K_{\text{Sea}}} F1_k \quad (6)$$

where the F1 score for each seen tag (k) is defined as:

$$F1_k = \frac{2P_k R_k}{P_k + R_k} \quad (7)$$

and

$$P_k = \frac{TP_k}{TP_k + FP_k}, R_k = \frac{TP_k}{TP_k + FN_k}. \quad (8)$$

where K_{Seen} is the number of distinct seen-tag classes; $F1_k$ is the F1 score of the k-th seen-tag class; P_k, R_k are precision and recall for the k-th seen-tag class, respectively; and TP_k, FP_k, FN_k are the number of true positives, false positives, and false negatives for the k-th seen-tag class, respectively.

To evaluate datasets across various model architectures and different numbers of parameters, we trained five different domain-adapted Transformer-based models for each dataset: BERT-base-uncased, RoBERTa-base, DistilBERT-base-uncased, PhoBERT-base, and LLaMA. To adapt LLaMA for the infilling task, we attached a token-classification head to the final hidden states. During training, a masked-token prediction objective was employed: a subset of input tokens was randomly replaced by a special mask token, and the model was fine-tuned to reconstruct the original tokens using cross-entropy loss. The prediction head consisted of a linear projection from the hidden dimension to the vocabulary space. The LLaMA configuration used in our experiments comprised three transformer layers and eight attention heads. All these models were trained from scratch, with 15% of input tokens randomly masked during training. We also continued training the pre-trained PhoBERT-base model to evaluate the performance of a pre-trained Vietnamese language model compared to training from scratch. By comparing our models, both trained from scratch and further fine-tuned, with LLMs under zero-shot prompting such as ChatGPT and Gemini, we assess the domain adaptation capabilities of these models, particularly in the context of administrative forms.

4.1. Experiment settings

We used the PyTorch package and the Transformers library to train the models. The experiments were conducted on three different types of GPUs, depending on device availability at the time: NVIDIA L4, Tesla T4, and NVIDIA A100-SXM4. All language models were trained using the AdamW optimizer with a learning rate of 5e-5, a batch size of 512, and around 200 epochs, without applying early stopping. Table 2 shows the number of parameters of the models.

Table 2. Comparison of different models across datasets (number of parameters, in millions).

Model	IL 1	IL 2	IL 3	IL 4
Gemini 1.5 Flash	Not publicly disclosed			
ChatGPT-4o Mini	Not publicly disclosed			
RoBERTa-base (ours)	23	23	24	25
DistilBERT-base-uncased (ours)	23	23	24	25
LLaMA (ours)	625	629	641	652
BERT-base-uncased (ours)	23	23	24	25
PhoBERT-base (ours)	135	135	136	137

The number of parameters for each model increases with the size of the dataset. As described in the previous section, the datasets contain an increasing number of document types and tags. Therefore, more parameters were needed to encode this information, resulting in an increased parameter count.

Since the exact parameter counts for ChatGPT-4o Mini and Gemini 1.5 Flash are not publicly disclosed, direct comparison is not possible. Among the five language models, LLaMA had the highest parameter count, increasing from 625M on IL 1 to 652M on IL 4, due to its complex architecture. In contrast, the other models, PhoBERT-base, BERT-base-uncased, DistilBERT-base-uncased, and RoBERTa-base had significantly smaller architectures. PhoBERT was approximately 4.5× smaller, while the other models were about 27× smaller compared to LLaMA.

We also conducted an experiment to compare training time and inference time across models. As described in panel B of Fig. 3, when using the same GPU to train with the same number of epochs, LLaMA took the longest training time across all datasets, followed by PhoBERT, while BERT, DistilBERT, and RoBERTa had approximate training time. The training details of Gemini and ChatGPT are not publicly disclosed; however, due to their massive architectures, they are predicted to require weeks or even months to train. Inference on test sets is also illustrated in panel A of Fig. 3: when using the same NVIDIA A100 GPU for inference, the inference time of our models is approximately one to two minutes, while the response latency of the ChatGPT-4o Mini and Gemini 1.5 Flash APIs can take up to two to eight hours, depending on the dataset. While

these two measurements reflect different settings, the lower local inference time of our models demonstrates the efficiency of our approach for local inference.

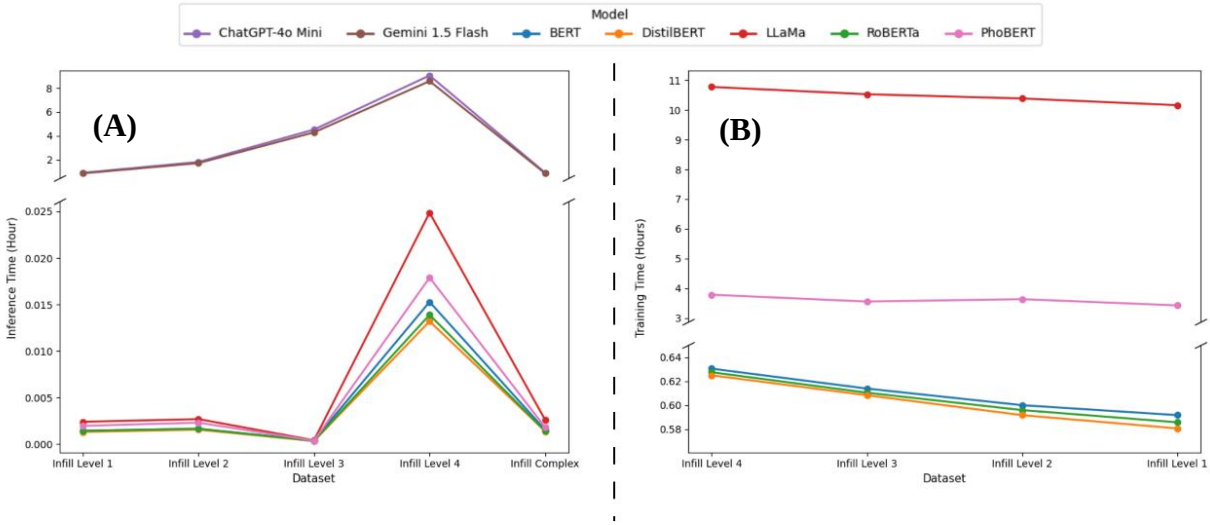


Fig. 3. Training and inference times of the language models across multiple datasets, evaluated on the test set using an A100 GPU. (A) Comparison of BERT, RoBERTa, LLaMA, DistilBERT, and PhoBERT inference time versus ChatGPT-4o Mini and Gemini 1.5 Flash API latency response time; **(B)** Comparison of training time for BERT, RoBERTa, LLaMA, DistilBERT, and PhoBERT.

4.2. Results on seen tags

Table 3 presents the accuracy scores and Macro F1 scores of various models across four datasets. Our models, particularly BERT-base-uncased and PhoBERT-base, demonstrate consistently strong performance, often surpassing two widely used LLMs under zero-shot prompting: ChatGPT-4o Mini and Gemini 1.5 Flash.

Table 3. Accuracy and Macro F1 comparison of models across datasets.

Metric	**Accuracy↑**				**Macro F1↑**			
Model	***IL 1***	***IL 2***	***IL 3***	***IL 4***	***IL 1***	***IL 2***	***IL 3***	***IL 4***
Gemini 1.5 Flash	0.8637	0.7825	0.7176	0.7349	0.8786	0.8070	0.7531	0.7816
ChatGPT-4o Mini	0.8625	0.7772	0.6765	0.6968	0.8642	0.7828	0.7221	0.7434
RoBERTa-base (ours)	0.7100	0.7597	0.7255	0.8367	0.7025	0.7549	0.6881	0.8508
DistilBERT-base-uncased (ours)	0.7752	0.6230	0.7336	0.7990	0.7664	0.5875	0.7031	0.8061

LLaMA (ours)	0.6635	0.5881	0.5900	0.6085	0.7356	0.6485	0.6472	0.6706
BERT-base-uncased (ours)	0.8647	0.8450	0.7093	0.8459	0.8595	0.8478	0.8498	0.8614
PhoBERT-base (ours)	0.8916	0.8653	0.8259	0.8389	0.8951	0.8149	0.8277	0.8533

Specifically, PhoBERT-base achieved the highest accuracy across IL 1-IL 3, with scores of 0.8916, 0.8653, and 0.8259, respectively, while BERT-base-uncased achieved the highest accuracy on IL 4, at 0.8459. Regarding Macro F1, our PhoBERT-base model outperformed both Gemini 1.5 Flash and ChatGPT-4o Mini across all four datasets. Our BERT-base-uncased model also outperformed both LLMs on IL 2-IL 4, although its Macro F1 score on IL 1 was lower than that of Gemini 1.5 Flash. Moreover, while the performance of ChatGPT and Gemini generally declines from IL 1 to IL 3 before recovering on IL 4, our models exhibit comparatively stable performance across the datasets. This indicates that domain-adapted Transformer-based models maintain a better understanding of context as content complexity and the number of tags increase, demonstrating both strong effectiveness and robustness in domain adaptation.

For example, a common paragraph that appears in many administrative forms is: “Toi ten la <RequesterName>, sinh ngay <DateOfBirth>, dia chi tam tru la <TemporaryA -ddress>...”. All our language models can distinguish between the temporary and permanent addresses and predict them correctly. However, the two LLMs, ChatGPT-4o Mini and Gemini 1.5 Flash, sometimes selected <PermanentAddress> instead of <TemporaryAddress> when both tags are provided, and vice versa. To be specific, our BERT-base-uncased model achieves an F1 score of 0.8227 on a common tag such as <PermanentAddress> on the IL 4 test set, outperforming ChatGPT, which achieves an F1 score of 0.8138. On the other hand, for a low-frequency tag such as <Profession>, our BERT-base-uncased model still achieves a higher F1 score of 0.32, compared to 0.1667 for ChatGPT. Moreover, we found no case in which ChatGPT-4o Mini or Gemini 1.5 Flash produced a correct prediction that our models could not. This example illustrates the potential benefit of domain adaptation for distinguishing semantically similar administrative tags in the constrained tag infilling setting.

In the dataset of administrative forms, several tags convey similar meanings, requiring the model to possess a nuanced understanding of language to distinguish

among them. Our results show that, although ChatGPT-4o Mini and Gemini 1.5 Flash are generally capable of making correct predictions, they still occasionally confuse semantically similar tags, as illustrated in Fig. 4. In contrast, our models demonstrate a stronger ability to accurately differentiate between such tags.

Specifically, among the 450 occurrences of the tag <FatherName>, BERT-base-uncased misclassified it as <MotherName> only once and <GuardianName> twice. In contrast, ChatGPT-4o Mini misclassified it as <GuardianName> 22 times and as <ParentFullName> once. These results confirm that our models maintain considerably lower confusion rates, enhancing reliability in infilling cases involving similar tags.

(A) **(B)**

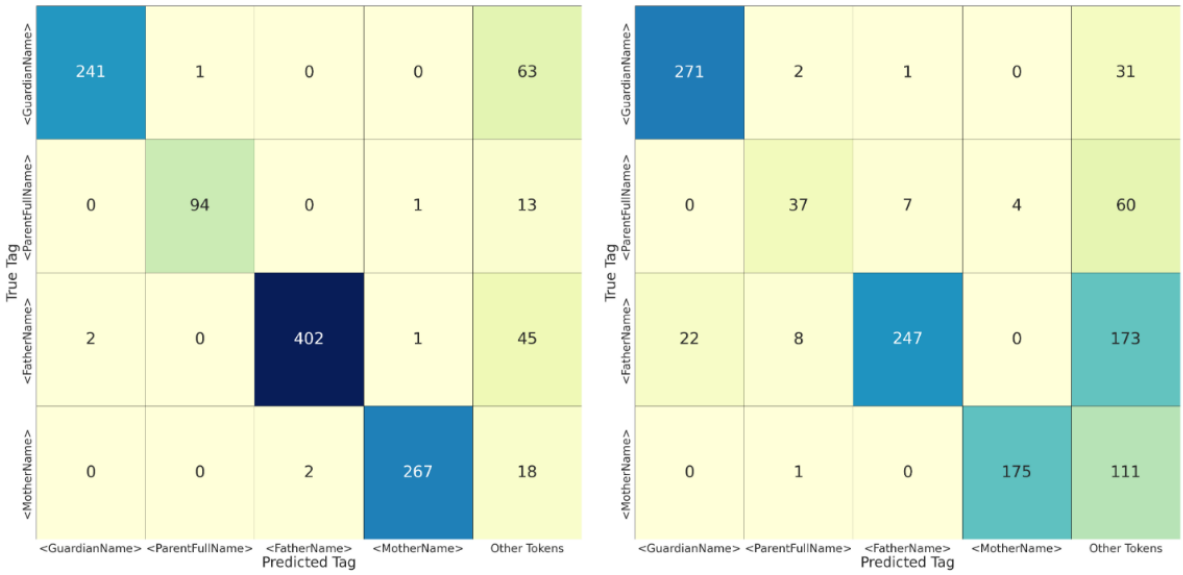


Fig. 4. Confusion matrices for a group of semantically similar tags. (A) Results from the BERT-base-uncased model; **(B)** Results from ChatGPT-4o Mini.

A similar trend is observed with the tag <ParentFullName>, which appeared 108 times. BERT-base-uncased misclassified it as <MotherName> once, whereas ChatGPT-4o Mini misclassified it as <FatherName> seven times and <MotherName> four times. This pattern is consistent across other semantically related tags, such as <MotherName> and <GuardianName>, highlighting the discriminative capability of our models in handling subtle semantic distinctions. Overall, these findings indicate that our domain-adapted Transformer-based models can provide more accurate predictions than the zero-shot ChatGPT-4o Mini and Gemini 1.5 Flash for several semantically similar administrative tags,

highlighting the potential of domain adaptation for improving semantic precision in administrative tag infilling.

4.3. Our model effectively flagged unseen tags

Administrative documents often contain numerous rare tags that appear only once and are specific to a particular form. To address this challenge, we demonstrate that our models can recognise previously unseen tags based on the probability values they assign to seen and unseen tags. As described in the previous section, the test set of Infill Complex includes several tags that do not appear in the training set. These unseen tags are used to evaluate the model's confidence when encountering unfamiliar information.

As illustrated in Fig. 5, the predicted probabilities for seen tags are typically above 50%. In contrast, for unseen tags, the model tends to produce probabilities below 50%, suggesting that it can effectively distinguish between known and unknown tag types. Moreover, we performed a Wilcoxon rank-sum test, which showed that, for each of our five language models, the predicted probability distributions of seen and unseen tags differed significantly ($p<0.05$).

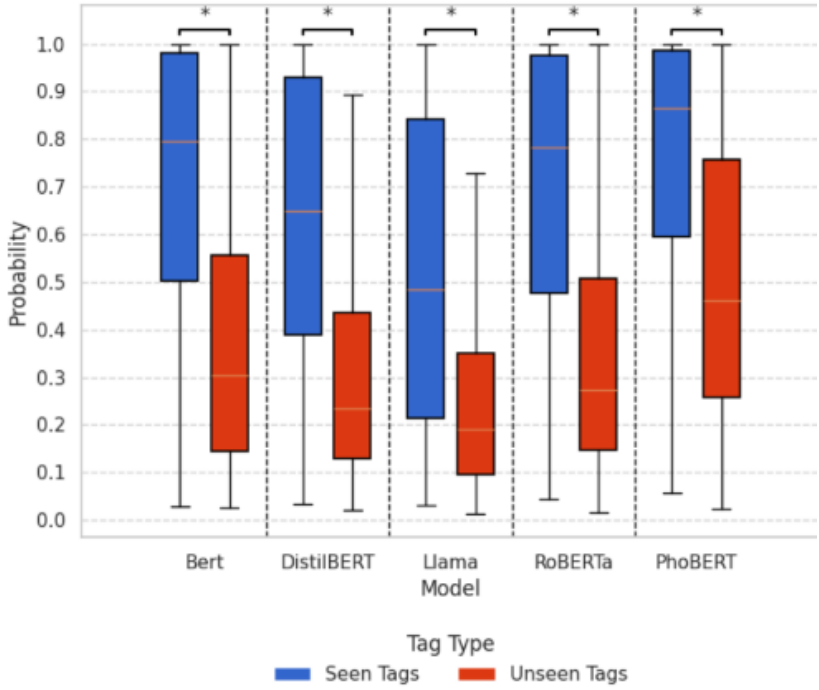


Fig. 5. Comparison of predicted probabilities between seen and unseen tags across models. Significance level: $p<0.05$ (*); determined by two-sided Wilcoxon rank-sum test.

Table 4 presents the accuracy of seen and unseen tag prediction on the Infill Complex dataset under different probability thresholds. As the threshold increases, the accuracy of unseen-tag detection generally improves, while the accuracy on seen tags decreases. This behaviour suggests that the models assign higher confidence scores to tags that were observed during training, whereas unseen tags tend to receive lower confidence scores.

Table 4. Accuracy of seen and unseen tag prediction under different probability thresholds on the Infill Complex dataset.

Infill Complex										
Accuracy↑										
Threshold	**0.3**		**0.4**		**0.5**		**0.6**		**0.7**	
Model	***Unseen***	***Seen***	***Unseen***	***Seen***	***Unseen***	***Seen***	***Unseen***	***Seen***	***Unseen***	***Seen***
ChatGPT-4o Mini	0.0187	0.5907	0.0187	0.5907	0.0187	0.5907	0.0187	0.5907	0.0187	0.5907
Gemini 1.5 Flash	0.3212	0.4898	0.3212	0.4898	0.3212	0.4898	0.3212	0.4898	0.3212	0.4898
BERT-base-uncased (ours)	0.4971	0.6431	0.6115	0.6258	0.7065	0.6017	0.7837	0.5603	0.8452	0.5178
DistilBERT-base-uncased (ours)	0.5994	0.7090	0.7175	0.6737	0.8048	0.6238	0.8712	0.5587	0.9156	0.4939
LLaMA (ours)	0.6946	0.4754	0.7921	0.4458	0.8602	0.4085	0.9032	0.3755	0.9296	0.3337
PhoBERT-base (ours)	0.5420	0.6797	0.5420	0.6702	0.5420	0.6503	0.6328	0.6058	0.7099	0.5574
RoBERTa-base (ours)	0.5361	0.6642	0.6548	0.6463	0.7435	0.6182	0.8156	0.5790	0.8707	0.5379

A threshold of 50% provides a reasonable trade-off between seen-tag and unseen-tag performance across most models, which is consistent with the probability distributions shown in Fig. 5. For unseen tags specifically, the LLaMA model achieved the highest accuracy at 0.8602 and the second highest was DistilBERT-base-uncased, which had 0.8048. For seen tags, the PhoBERT-base model achieved the highest accuracy at 0.6503, and the second highest was DistilBERT-base-uncased, which achieved 0.6238. These results demonstrate that several domain-adapted Transformer-based models can achieve strong performance in both seen-tag prediction and unseen-tag detection.

For Gemini 1.5 Flash and ChatGPT-4o Mini, threshold ablation could not be performed because token-level prediction probabilities are not publicly available through their APIs. Nevertheless, the results show that the proposed domain-adapted Transformer-based models outperform these zero-shot LLMs on most thresholds for both seen and unseen tags, except at the highest threshold (70%) for seen-tag prediction.

4.4. Our model could avoid overconfident predictions

As mentioned in the introduction section, one of the disadvantages of LLMs is their tendency to exhibit overconfidence in their predictions. This phenomenon is illustrated in Fig. 6, which compares the predictions made by ChatGPT-4o Mini and our BERT-base-uncased model for three unseen tags and four seen tags with similar semantic meanings.

In detail, despite not being provided with these unseen tags during training, ChatGPT-4o Mini frequently misclassified them as size-related dimension tags. For instance, it predicted <TotalFloorArea> as <SizeLength> 222 times, <SizeWidth> 43 times, and <SizeHeight> 9 times. Similarly, it misclassified <NumberOfFloors> as <SizeHeight> 55 times, <SizeWidth> 7 times, and <SizeLength> 3 times. A similar pattern was observed for <ConstructionLevel>, which was incorrectly predicted as <SizeLength> 12 times, <SizeHeight> 6 times, and <SizeWidth> 3 times. These outcomes indicate that ChatGPT-4o Mini struggles to differentiate conceptually distinct tags outside its training scope, often defaulting to more familiar categories.

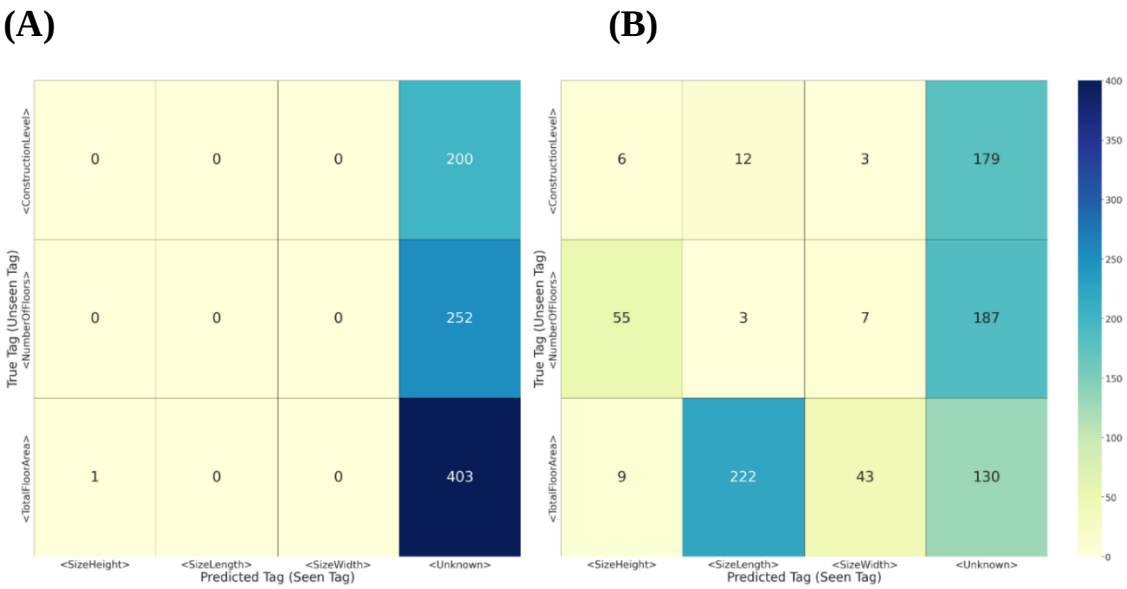


Fig. 6. Confusion matrices for a group of seen tags that are semantically similar to unseen tags. (A) Results from the BERT-base-uncased model; **(B)** Results from ChatGPT-4o Mini.

In contrast, our model BERT-base-uncased demonstrated significantly better performance. It correctly predicted all instances of <ConstructionLevel> and <NumberOfFloors>, and made only a single misclassification for <TotalFloorArea>, predicting it once as <SizeHeight>. These results suggest that our models generalise more reliably to unseen tags and do not exhibit the same overconfidence as prompting ChatGPT-4o Mini when encountering out-of-domain data. This indicates that our domain-adapted Transformer-based models can more reliably flag unseen tags and avoid overconfident assignment of unseen cases to familiar tags.

5. Conclusions

In this article, we introduce a comprehensive and scalable pipeline for Vietnamese administrative tag infilling, formulated as a constrained tag prediction task for classifying administrative document tags. The pipeline includes steps such as data collection, data creation, model training, and testing. It forms a comprehensive system capable of filling in information for a variety of forms generated from authentic Vietnamese administrative documents. In addition, we demonstrate that our domain-adapted Transformer-based language models can outperform ChatGPT-4o Mini and Gemini 1.5 Flash when the latter are evaluated using zero-shot prompting in the constrained administrative tag prediction setting,

despite having thousands of times fewer parameters. Interestingly, we also find that training models from scratch can yield better performance than pre-trained Vietnamese language models like PhoBERT on certain metrics, such as Macro F1. This may be due to the fact that pre-trained models carry inherent biases from their original training data, whereas models trained from scratch can adapt more effectively to the new dataset. Furthermore, our models showed the ability to flag instances of tags not present in the training dataset by assigning lower confidence to their predictions, without identifying the specific unseen tag. These findings highlight the potential of domain-adapted Transformer-based models to provide an effective and computationally efficient solution for Vietnamese administrative tag infilling.

We also found that there is a difference in performance among Transformer family models. For example, BERT-base-uncased achieved higher accuracy in most datasets with seen tags, especially compared with RoBERTa, a model often known for its better performance than BERT. However, this behaviour is not rare; it was previously observed in the work of T. Shin, et al. (2020) [19], where the authors performed validation between models using similar fill-in-the-blank tasks. Their experiments also showed that in low-data settings, BERT outperforms RoBERTa in accuracy. This setting is similar to our datasets, each of which has a modest number of training samples compared with a typical language-model dataset. We also noticed that the DistilBERT model had significantly better performance on unseen tags. As stated by V. Sanh, et al. (2019) [20], DistilBERT uses a triple-loss objective for knowledge distillation, which might act as a form of regularisation that increases generalisation and helps it distinguish between seen and unseen tags in this case.

In addition to these observations, we also identified several limitations in this study. First, the data used in the research was generated by ChatGPT-4o Mini. As mentioned in previous sections about ChatGPT, this type of data has certain limitations, such as lower accuracy compared to human-generated data and the inability to create new discoveries, only replicating patterns. Although we filtered low-quality paragraphs using a Vietnamese dictionary, this approach cannot detect cases where words are correct but used in an incorrect context. These factors may introduce minor inconsistencies when the model is deployed in real-world scenarios. Moreover, within the scope of the study, we only applied the

method to about 50 Vietnamese administrative forms, a modest number compared to the hundreds of forms currently in use. Therefore, a potential future research direction is to increase the number of forms used in the study to further demonstrate the potential of this approach.

Second, we observed consistently low performance from the LLaMA model across IL 1-4. This underperformance may be due to overfitting, potentially caused by its complex architecture and limited fine-tuning data. However, further experiments are necessary to verify this hypothesis. Finally, the comparison is conducted between zero-shot prompting with LLMs and domain-adapted Transformer-based models trained on the dataset. This raises the question of whether few-shot prompting or fine-tuning LLMs could further improve performance. Therefore, our results should be interpreted specifically as a comparison between domain-adapted Transformer-based models and zero-shot LLMs prompting in the constrained administrative tag infilling setting, rather than as a definitive comparison between them. We also note that attention-based CNN models would be valuable baselines for future studies because they have demonstrated strong performance in document-specific and domain-specific text classification tasks [21]. Future work addressing these limitations could further establish the applicability of the proposed approach as a practical tool for administrative form processing.

CRediT author statement

Vinh Nguyen: Conceptualisation, Methodology, Data curation, Formal analysis, Investigation, Validation, Visualisation, Writing original draft; Tung Le: Supervision, Writing - Review and Editing; Huy Tien Nguyen: Conceptualisation, Methodology, Supervision, Validation, Writing - Review and Editing.

COMPETING INTERESTS

The authors declare that there is no conflict of interest regarding the publication of this article.

REFERENCES

[1] A.R. Salama, Ö. Alaçam, W. Menzel (2018), “Text completion using context-integrated dependency parsing”, *Proceedings of The Third Workshop on Representation Learning for NLP*, pp.41-49, DOI: 10.18653/v1/W18-3005.

[2] P. Mirowski, A. Vlachos (2015), "Dependency recurrent neural language models for sentence completion", *Proceedings of The 53rd Annual Meeting of The Association for Computational Linguistics and The 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp.511-517, DOI: 10.3115/v1/P15-2084.

[3] M. Zhang, M. Huang, R. Shi, et al. (2024), "Calibrating the confidence of large language models by eliciting fidelity", *Proceedings of The 2024 Conference on Empirical Methods in Natural Language Processing*, pp.2959-2979, DOI: 10.18653/v1/2024.emnlp-main.173.

[4] J. Leng, C. Huang, B. Zhu, et al. (2025), "Taming overconfidence in LLMs: Reward calibration in RLHF", *13th International Conference on Learning Representations (ICLR 2025)*, https://proceedings.iclr.cc/paper_files/paper/2025/file/29fb6e1456b3d8b57ede5c45aa2c6537-Paper-Conference.pdf, accessed 20 September 2026.

[5] C. Xu, B. Wen, B. Han, et al. (2025), "Do language models mirror human confidence? Exploring psychological insights to address overconfidence in LLMs", *Findings of The Association for Computational Linguistics: ACL 2025*, pp.25655-25672, DOI: 10.18653/v1/2025.findings-acl.1316.

[6] A. Hamdi, M. Coustaty, A. Joseph, et al. (2018), "Feature selection for document flow segmentation", *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pp.245-250, DOI: 10.1109/DAS.2018.66.

[7] S. Biswas, P. Riba, J. Lladós, et al. (2021), "Graph-based deep generative modelling for document layout generation", *Document Analysis and Recognition - ICDAR 2021 Workshops*, pp.525-537, DOI: 10.1007/978-3-030-86159-9_38.

[8] X. Ling, M. Gao, D. Wang (2020), "Intelligent document processing based on RPA and machine learning", *2020 Chinese Automation Congress (CAC)*, pp.1349-1353, DOI: 10.1109/CAC51589.2020.9326579.

[9] Z.A. Nazi, M.R. Hossain, F.A. Mamun (2025), "Evaluation of open and closed-source LLMs for low-resource language with zero-shot, few-shot, and chain-of-thought prompting", *Natural Language Processing Journal*, **10**, DOI: 10.1016/j.nlp.2024.100124.

[10] H. Gurdil, Y.B. Soguksu, S. Salihoglu, et al. (2025), "Integration of artificial intelligence in educational measurement: Efficacy of ChatGPT in data generation within the scope of item response theory", *Trakya Eğitim Dergisi*, DOI:10.24315/tred.1509299 (in Turkish).

[11] H. Tari, M.D. Khan, J. Rutten, et al. (2024), "Leveraging GPT for the generation of multi-platform social media datasets for research", *HT'24: Proceedings of The 35th ACM Conference on Hypertext and Social Media*, pp.337-343, DOI: 10.1145/3648188.3675153.

[12] S.A. Lehr, A. Caliskan, S. Liyanage, et al. (2024), "ChatGPT as research scientist: Probing GPT's capabilities as a research librarian, research ethicist, data generator, and data predictor", *Proceedings of The National Academy of Sciences*, **121(35)**, DOI: 10.1073/pnas.2404328121.

[13] K. Song, X. Tan, T. Qin, et al. (2019), "MASS: Masked sequence to sequence pre-training for language generation", *Proceedings of The 36th International Conference on Machine Learning (ICML)*, PMLR **97**, pp.5926-5936.

[14] M. Lewis, Y. Liu, N. Goyal, et al. (2020), "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension", *Proceedings of The 58th Annual Meeting of The Association for Computational Linguistics*, pp.7871-7880, DOI: 10.18653/v1/2020.acl-main.703.

[15] W. Zhu, Z. Hu, E. Xing (2019), "Text infilling", *ArXiv Preprint*, DOI: 10.48550/arXiv.1901.00158.

[16] C. Donahue, M. Lee, P. Liang (2020), "Enabling language models to fill in the blanks", *Proceedings of The 58th Annual Meeting of The Association for Computational Linguistics*, pp.2492-2501, DOI: 10.18653/v1/2020.acl-main.225.

[17] D. Li, J. You, K. Funakoshi, et al. (2022), "A-TIP: Attribute-aware text infilling via pre-trained language model", *Proceedings of The 29th International Conference on Computational Linguistics*, pp.5857-5869.

[18] H.N. Duc (2003), *Vietnamese Dictionary*, https://github.com/undertheseanlp/dictionary/tree/hongocduc (in Vietnamese).

[19] T. Shin, Y. Razeghi, R.L. Logan IV, et al. (2020), "AutoPrompt: Eliciting knowledge from language models with automatically generated

prompts", *Proceedings of The 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp.4222-4235, DOI: 10.18653/v1/2020.emnlp-main.346.

[20] V. Sanh, L. Debut, J. Chaumond, et al. (2019), "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter", *5th Workshop on Energy Efficient Machine Learning and Cognitive Computing @ NeurIPS 2019*.

[21] H.T. Nguyen, M.K. Phi, X.B. Ngo, et al. (2024), "Attentive deep neural networks for legal document retrieval", *Artificial Intelligence and Law*, **32(1)**, pp.57-86, DOI: 10.1007/s10506-022-09341-8.