Evaluation of Topic Continuity Using Nonlinearlized Naive Bayes With Attention Mechanism

Shu-Ting Pi*
shutingp@amazon.com
Amazon
Cupertino, CA, USA

Pradeep Bagavan*
deepbag@amazon.com
Amazon
New York, NY, USA

Yejia Li
imyejia@amazon.com
Amazon
Seattle, WA, USA

Disha
Disha@amazon.com
Amazon
Seattle, WA, USA

Qun Liu
qunliu@amazon.com
Amazon
Seattle, WA, USA

ABSTRACT

Utilizing Large Language Models (LLM) as chatbots in diverse business scenarios often presents the challenge of maintaining topic continuity. Abrupt shifts in topics can lead to poor user experiences and inefficient utilization of computational resources. In this paper, we present a topic continuity model aimed at assessing whether a response aligns with the initial conversation topic. Our model is built upon the expansion of the corresponding natural language understanding (NLU) model into quantifiable terms using a Naive Bayes approach. Subsequently, we have introduced an attention mechanism and logarithmic nonlinearity to enhance its capability to capture topic continuity. This approach allows us to convert the NLU model into an interpretable analytical formula. In contrast to many NLU models constrained by token limits, our proposed model can seamlessly handle conversations of any length with linear time complexity. Furthermore, the attention mechanism significantly improves the model's ability to identify topic continuity in complex conversations. According to our experiments, our model consistently outperforms traditional methods, particularly in handling lengthy and intricate conversations. This unique capability offers us an opportunity to ensure the responsible and interpretable use of LLMs.

1 INTRODUCTION

The rise of large-scale language models (LLMs) [1, 2] has empowered chatbots to handle various business tasks, such as serving as office assistants [3], coding companions [4, 5], and data explorers [6]. However, leveraging LLMs for these roles often presents challenges like hallucination [7], offensive language [8], prompt injection [9], and adversarial attacks [10]. In addition to these common issues, specific business applications may introduce unique problems, such as maintaining topic continuity. For example, when using LLMs as a customer service chatbot, LLMs are employed to address inquiries about specific products or services. However, because LLM responses are inherently random, there's no guarantee that they will consistently remain focused on the intended topics, potentially resulting in a subpar user experience. On the other hand, if users veer off into unrelated topics, it could also lead to the waste of valuable computational resources. Therefore, ensuring topic coherence between the customer and the chatbot is crucial.

In customer service, users initially describe their concerns. When these concerns pertain to the business's operations, the customer and chatbot collaborate on solutions [11–14]. Ensuring a smooth conversation involves assessing if the current sentence logically follows the prior ones. For example, if a user discussing refunds suddenly asks, "Can you help me order a pizza?" – it's off-topic. This concept is formalized as a natural language understanding model (NLU) [15], denoted as $P(y|S_1, S_2, ...; S_N)$. Here, S_i (for $i = 1$ to $N-1$) represents previous N-1 sentences, and S_N is the current one. The binary variable y indicates whether S_N aligns with preceding sentences, keeping the conversation on-topic.

In practical use, when users interact with LLM, we assess if each new sentence, whether from the user or the LLM, keeps the conversation on-topic. If it goes off-topic, we guide it back to business-related subjects or may end the conversation. So, we assume the previous N-1 sentences are on-topic, and we calculate whether the newly added N_{th} sentence still aligns with the ongoing conversation. **This simplifies the problem to determining whether the N_{th} sentence has a reasonable contextual relationship with the previous N-1 sentences**. The most commonly used approach to address this issue is a BERT-based language model [16, 17]. These models are inherently equipped with the capability to evaluate the contextual relationship between two sentences. However, employing this approach consistently gives rise to two inevitable challenges: **1) Token Size Limit** and **2) Lack of Sentence Attention**.

*Both authors contributed equally

Regarding the first challenge, imagine using a language model to assess the connection between $(S_1+S_2+\ldots+S_{N-1})$ and the current sentence S_N in a conversation. As the conversation grows, the text often exceeds most language models' token limits, typically set at 512 tokens for many BERT-based models. Regarding the second challenge, most language models are trained on sentence pairs from articles where semantic relationships are consistently close. However, real conversations often involve looser semantic connections. For example, a customer might say, "Earlier, you asked about the missing product serial number, but now I've found it." This response references a part of the conversation from several rounds back. Concatenating $S_1 \sim S_{N-1}$ as context can lead to the model struggling to judge the appropriateness of S_N as a follow-up. In summary, an effective conversational topic continuity model must address two key challenges: **1) handling lengthy conversations**, and **2) accommodating semantic leaps**.

To address these challenges, we introduce an innovative topic continuity model that integrates logarithmic nonlinearity and sentence attention into the naive Bayes framework [18]. Our method provides a fully analytical formulation of the problem, effectively addressing the aforementioned issues and delivering significantly superior performance compared to conventional methods.

2 NOLINEAR NAIVE BAYES WITH ATTENTION MECHANISM

2.1 Model Definition

When a user is engaged in a conversation with a chatbot, our goal is to identify topic shifts in new sentences, **assuming that the first N-1 sentences are on-topic**. As discussed in Section 1, we can define an NLU model for this problem as a conditional probability expressed as follows:

$$P(y|S_1, S_2, \ldots; S_N) \tag{1}$$

, where $S_1 \sim S_{N-1}$ represents the previous $N-1$ sentences, S_N represents the current sentence, and y, a binary variable, signals whether the text composed of $S_1, \sim S_N$ deviates from the topic. In fact, we can broaden the interpretation of each variable in Eq.(1). S_i need not be limited to single sentences; it can also encompass chunks of multiple sentences, potentially with overlapping content, as long as the relationships between S_i maintain sentence information and sequence. Our research indicates that employing a sliding window with appropriate size and strides to construct sentence chunks consistently yields the best results. **Hence, unless specified otherwise, we assume that all $S_i, i = 1 \sim N-1$, represent sentence chunks, with S_N being a single sentence.**

2.2 Naive Bayes With Attention

While estimating Eq.(1) directly using models like BERT is possible, this approach presents the two issues outlined in Section 1. To address these challenges, let's begin with the Naive Bayes assumption, where the variables $(S_1, \ldots; S_N)$ are considered independent of each other, and we expand Eq.(1) upon this assumption as follows:

$$\begin{aligned} P(y|S_1, S_2, \cdots; S_N) &= \frac{P(S_1|y)P(S_2|y)\cdots P(S_N|y)P(y)}{P(S_1)P(S_2)\cdots P(S_N)} \\ &= \Pi_i^N \left[\frac{P(S_i|y)}{P(S_i)}\right] P(y) \end{aligned} \tag{2}$$

Indeed, the Naive Bayes assumption that there is no semantic connection between sentences contradicts the core problem addressed in this paper. Therefore, we utilize Naive Bayes purely as a mathematical tool in this context and we will introduce additional techniques to overcome the limitations inherent in the Naive Bayes assumption.

We aim to incorporate an attention mechanism into Eq.(2). To achieve this, we have intentionally reformulated the equation to include pairwise probability. Consequently,

$$P(y|S_i, S_N) = \frac{P(S_i, S_N|y)P(y)}{P(S_i, S_N)} = \frac{P(S_i|y)P(S_N|y)P(y)}{P(S_i)P(S_N)}$$

Thus,

$$P(S_i|y) = \frac{P(y|S_i, S_N)P(S_i)P(S_N)}{P(S_N|y)P(y)}$$

Let's plug this term into Eq.(2). We have,

$$P(y|S_1 \ldots; S_N) = \Pi_i^N \left\{\frac{P(y|S_i, S_N)P(S_i)P(S_N)}{P(S_N|y)P(y)} \frac{1}{P(S_i)}\right\} P(y)$$

So,

$$P(y|S_1 \cdots; S_N) = \Pi_i^N \{P(y|S_i, S_N)\} P^{-N}(S_N|y)P^N(S_N)P^{1-N}(y)$$

Take log on both side,

$$\begin{aligned} logP(y|S_1 \cdots; S_N) = &\sum_{i=1}^{N}\{logP(y|S_i, S_N)\} \\ &- NlogP(S_N|y) + NlogP(S_N) + (1-N)logP(y) \end{aligned}$$

Note that in the first summation, there exists a term $logP(y|S_N, S_N)$, which can be approximated as $logP(y|S_N, S_N) \approx logP(y|S_N) = logP(S_N|y) + logP(y) - logP(S_N)$. Additionally, the term $logP(y)$ is essentially a constant and does not affect any of the subsequent calculations, so we can safely disregard this term. Thus, we have:

$$\begin{aligned} logP(y|S_1 \cdots; S_N) = &\sum_{i=1}^{N-1} \{logP(y|S_i, S_N)\} \\ &+ (N-1)\left[logP(S_N) - logP(S_N|y)\right] \end{aligned} \tag{3}$$

The equation above has several key points. Firstly, we introduced a pairwise term for chunk/current-sentence pairs, directing attention from the current sentence, S_N, to another chunk, S_i. Secondly, expressing Naive Bayes in logarithmic probabilities simplifies the problem, yielding a linear outcome. Lastly, each term involves a maximum of one chunk plus one sentence, ensuring token length stays within language model limits. As the conversation progresses, time consumption increases linearly, but deep learning models can batch attention terms, potentially maintaining constant time consumption if the chunk count remains within GPU memory limits.

2.3 Logarithmic Non-linearity

As discussed in the previous section, the assumption of independent variables, leading to a linear combination of logarithmic terms, is inadequate for addressing this problem. Therefore, we need to make Eq.(3) nonlinear to overcome the limitations of Naive Bayes.

To introduce nonlinearity, let's analyze each term. In Eq. (3), the first term computes an equal-weighted average among the attention terms, omitting the factor $1/(N-1)$. This operation resembles a mathematical "functional," transforming the vector $[logP(y|S_i, S_N), i = 1 \sim N-1]$ into a single scalar value. In machine learning, this is often referred to as average pooling.

Regarding the second term, comprised of $[logP(S_N) - logP(S_N|y)]$, its meaning is straightforward. Let's consider a customer service chatbot scenario where the user's focus is solely on a specific product, like a cell phone. Here, $logP(S_N|y)$ represents the likelihood of sentence S_N occurring within this product-specific context, while $logP(S_N)$ represents the log-probability of sentence S_N appearing in any chatbot conversation without specific product restrictions. Therefore, a more negative value on this term highlights the likelihood of the sentence S_N being more focused on the topic of cell phones.

Based on the above discussion, a straightforward approach is to maintain the mathematical form but introduce more non-linear operations. This can be achieved by replacing $\sum \rightarrow \mathcal{F}$ and $(N-1) \rightarrow \alpha$ as shown below:

$$logP(y|S_1 \cdots S_N) = \mathcal{F}\left\{logP(y|\tilde{\mathbf{S}}, S_N)\right\} + \alpha(\mathbf{S})\,[logP(S_N) - logP(S_N|y)] \quad (4)$$

, where $logP(y|\tilde{\mathbf{S}}, S_N)$ is a vector composed of $logP(y|S_i, S_N)$ with $i = 1 \sim N-1$, $\mathcal{F}$ is an arbitrary functional that transforms the vector into a scalar, and α, is a positive coefficient (since $N-1 > 0$) dependent on all sentence chunks, including S_N. In Eq.(4), we've replaced the original equal-weighted averaging on $logP(y|\tilde{\mathbf{S}}, S_N)$ with a custom functional $\mathcal{F}$ and transformed the coefficient in the second term into functions related to $\mathbf{S}$. Although Eq.(4) resembles Eq.(3), **it no longer relies on the independence variable assumption of naive Bayes**. We'll refer to the first term as the "attention term" and the second term as the "residual term", highlighting the difference between two log-probabilities. In the upcoming section, we'll delve into the design of $\mathcal{F}$ and α.

3 FORMULATION OF NONLINEAR TRANSFORMATION

3.1 Designing Attention Functional

In a conservation, sentences typically fall into three scenarios: **1). Normal Sentences** correspond to responses to the previous sentence, the most frequent scenario. **2). Leap Sentences** correspond to responses to earlier sentences in the conversation, constituting a "leap conversation". In the following, we use the term "*target sentence*" to denote the sentence that the current sentence S_N responds to. **3). Topic Shift Sentences** indicate a shift in topic.

To capture these three scenarios, we define the notation $log\mathcal{P}_{max} = \max\{logP(y|\tilde{\mathbf{S}}, S_N)\}$ and $log\mathcal{P}_{avg} = avg\{logP(y|\tilde{\mathbf{S}}, S_N)\}$. Then the attention functional is defined as:

$$\mathcal{F}\left\{logP(y|\tilde{\mathbf{S}}, S_N)\right\} = [1 + \tanh(log\mathcal{P}_{max})]\, log\mathcal{P}_{max} - \tanh(log\mathcal{P}_{max}) log\mathcal{P}_{avg} \quad (5)$$

As log-probabilities are always negative, the first coefficient, $1 + tanh(log\mathcal{P}_{max})$, indicates that as $log\mathcal{P}_{max}$ approaches zero, we primarily use $log\mathcal{P}_{max}$ to approximate Eq.(1). Conversely, as $log\mathcal{P}_{max}$ approaches negative infinity, we rely on $log\mathcal{P}_{avg}$ for the estimate.

The approach is clear. In Scenario 1, assuming previous text $S_1, \ldots S_{N-1}$ is on-topic and S_N responds to S_{N-1}, we focus on evaluating if S_N aligns with S_{N-1}, approximating $P(y|S_1, \ldots S_N) \approx P(y|S_{N-1}, S_N)$. Similarly, in Scenario 2, when S_N responds to a specific chunk earlier in the conversation, we expect $P(y|S_1, \ldots S_N) \approx P(y|S_{target}, S_N)$. In both scenarios, where there's a clear link between current sentences and a specific chunk, the likelihood they form often peaks in the $logP(y|\tilde{\mathbf{S}}, S_N)$ vector. Hence, for these cases, we choose $log\mathcal{P}_{max}$ as the dominant term.

When S_N abruptly changes topics, it lacks context within the conversation, leading to bias if using $log\mathcal{P}_{max}$ for Eq.(1). Instead, opting for $log\mathcal{P}_{avg}$ is better. In this scenario, Eq.(5) simplifies to the naive Bayes case, indicating that the independence variable assumption is a suitable approximation for the NLU model when there's no clear contextual link between the current sentence and prior conversation.

3.2 Designing Residual Coefficient

Our experiments consistently show that Eq.(5) often provides outstanding results on its own. Hence, when crafting the

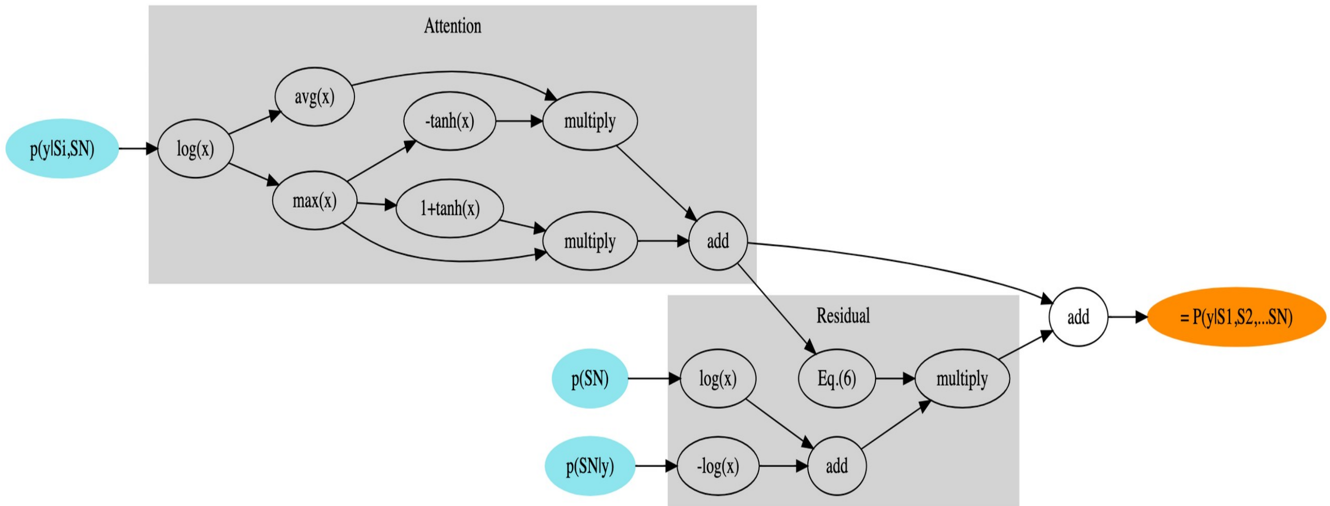

Figure 1: Computation graph for calculating the NLU likelihood (highlighted in orange). The blue blocks represent fundamental components of our model.

residual coefficient, we view it as a corrective purturbation for situations where Eq.(5) lacks confidence. By defining the probabilities $P_{nlu} = e^{P(y|S_1,...,S_N)}$ and $P_{att} = e^{\mathbf{F}\{P(y|\mathbf{S},S_N)\}}$ from the NLU model and attention term respectively, we aim for the perturbation to possess three key properties: 1) Peak at $P_{att} = 0.5$ (low confidence), 2) Approach zero as P_{att} nears 0.0 or 1.0 (high confidence), and 3) Be unbiased, symmetrical around $P_{att} = 0.5$.

To fulfill these criteria, a straightforward mathematical form is a sine function:

$$P_{nlu} = P_{att} + \beta \sin(\pi P_{att})$$

,where $\beta \ll 0.5$. The condition $\beta \ll 0.5$ arises from the situation where the perturbation term attains its maximum value at $P_{att} = 0.5$ and $P_{nlu} = 0.5 + \beta$. Given its nature as a perturbation, β must be $\ll 0.5$. By taking a logarithm on both side, we get:

$$\begin{aligned} logP_{nlu} &= log[P_{att} + \beta \sin(\pi P_{att})] \\ &= log(P_{att}) + log[1 + \beta sin(\pi P_{att})/P_{att}] \end{aligned}$$

. Since $\beta sin(\pi P_{att})/P_{att} \ll 1$, first order of Taylor expansion yields

$$logP_{nlu} \approx log(P_{att}) + \beta \sin(\pi P_{att})/P_{att}$$

.

Comparing this from with eq.(4), we assert α should be:

$$\alpha = \frac{\sin(\pi e^{\mathbf{F}\{P(y|\mathbf{S},S_N)\}})}{e^{\mathbf{F}\{P(y|\mathbf{S},S_N)\}}} \frac{\eta}{|log(\epsilon)|} \tag{6}$$

Here, P_{att} is represented as its original form $e^{\mathbf{F}\{P(y|\mathbf{S},S_N)\}}$ and the term $\eta/|log(\epsilon)|$ serves as a scaling factor with $\eta \ll 0.5$ and ϵ is an arbitrarily small number, such as 10^{-3} used in this article. The rationale behind the scaling factor is evident. As a probability P approaches 0, $logP$ approaches $-\infty$. Thus, in practical calculations, we designate a small value ϵ, and any probability lower than ϵ is set to ϵ to prevent computational instability. Consequently, the log-difference term $[\log P(S_N) - \log P(S_N|y)]$ in eq.(4) ranges between $\pm\log(\epsilon) \approx \pm 6.9$. By incorporating $|\log(\epsilon)|$ into the scaling factor, we normalize the log-difference to fall within the range of -1 to $+1$. Since

$$\beta = \frac{\eta}{|\log(\epsilon)|} [\log P(S_N|y) - \log P(S_N)] \ll 0.5$$

by comparing with eq.(6), it is imperative to ensure that $\eta \ll 0.5$.

Eq. (6) holds mathematical significance. $\sin(\pi P_{att})/P_{att}$ guarantees adherence to the three properties mentioned earlier. The log-difference $[\log P(S_N) - \log P(S_N|y)]$ in eq.(4) measures the perturbation's magnitude, normalized by $|log(\epsilon)|$, while η controls its maximum strength. Though derived from the perturbation assumption, eq.(6) ensures P_{nlu} stays within the 0 to 1 range, akin to a probability, as long as $\eta \leq 0.5$. **In the following, we stick to $\epsilon = 0.001$ and $\eta = 0.2$, usually yielding favorable outcomes, unless stated otherwise**.

3.3 Estimation of Fundamental Components

So far, we have derived all the expressions for NLU model, which are given by Eq.(4), Eq.(5), and Eq.(6). To compute these formulas, we need to estimate $P(y|S_i, S_N)$, $P(S_N|y)$, and $P(S_N)$.

Attention Term $P(y|S_i, S_N)$ involves determining whether there is a contextual relationship between (S_i, S_N), and this can be estimated using language models like BERT. In many machine learning papers, this task is often referred to as Next Sentence Prediction (NSP) [19, 20]. There are many open-source NSP models available on platforms like Hugging Face and there's no need for us to retrain them.

Residual Term Estimating $P(S_N|y)$ and $P(S_N)$ involves context-dependent factors. In theory, these quantities should be calculated through integration over all variables:

$$P(S_N|y) = \int P(S_1 \dots S_N|y) dS_1 \dots dS_{N-1}$$

and

$$P(S_N) = \int P(S_1 \dots S_N) dS_1 \dots dS_{N-1}$$

. However, practical calculations of these integrals are improbable. Instead, we employ an indirect approach.

For instance, consider a customer service chatbot designed to respond to various product-related queries, such as "cell phones." To establish $P(S_N|y)$ for the "cell phone" topic, we randomly sample numerous sentences from historical conversations with topic of cell phones. Estimating the likelihood of a sentence appearing in the context of the topic can be done using an out-of-distribution (OOD) method, like Isolation Forest [21, 22]. Here's how it works:

- Encode each sentence using a pre-trained models, such as Sentence BERT [23].
- Train an Isolation Forest with this dataset to generate anomaly scores for all sentences. Here we invert the sign compared to the original paper, so higher anomaly scores θ signify a greater likelihood of a sentence being included in the dataset.
- Once the distribution of θ is obtained, we estimate its probability density function $p(\theta)$ and for a future sentence with a score $\theta = c$, the corresponding probability is given by the Cumulative Distribution Function (CDF):

$$p(S_N|y) = \int_{-\infty}^{c} p(\theta) d\theta$$

.

We can use the same approach to estimate $P(S_N)$, but without specific topic constraints. For $P(S_N)$, we sample sentences from historical dialogue data across all topics to train the OOD model. In practical business scenarios, chatbots are often designed to answer questions related to limited product lines. Therefore, we can pre-train $p(S_N|y)$ for each product line and store them in cache. When a conversation's topic is determined, we swiftly employ the corresponding model.

Regarding the use of CDF as probabilities, it may seem that assigning a probability of 100% to data with the highest scores is unreasonable. However, our primary interest lies in the difference in log-probabilities. Therefore, as long as the hyperparameters of these two OOD models are similar enough to ensure that the anomaly score distributions they estimate fall within a comparable range, their differences remain meaningful for log-probabilities.

So far, we have approximated Eq.(1) using Eq.(4)-(6). To help readers understand the calculation process, we have represented a computation graph in Figure 1.

4 EXPERIMENTS

4.1 Dataset

For the experiment, we used a dataset that was generated by professional customer service agents interacting with an LLM, simulating customers asking the LLM questions related to online video streaming. The dataset was entirely generated through simulation and did not use any real user data, with the purpose of protecting user privacy. We sampled data from the following four different categories:

- **Normal Conversation (on-topic)** 1000 data points where the current sentence responses to the preceding sentence.
- **Leap Conversation (on-topic)** 1000 data points in which the current sentence is a response to an earlier sentence in the conversation.
- **Out-of-Domain Topic Shift (off-topic)**: 1000 data points where the current sentence diverges completely from the main topic and is entirely unrelated to Amazon's services.
- **In-domain Topic Shift (off-topic)** 1000 data points in which the current sentence diverges significantly from the main topic but remains relevant to Amazon's services.

We labeled all on-topic data points, including normal and leap conversations, as $y = 1$, while deviation conversations were labeled as $y = 0$, resulting in a binary classification task. Note that while there are 4000 data points in the dataset, we don't need to collect 4000 distinct conversations (even though we have done so). We can simply choose different sentences as S_N within a single conversation to generate multiple data points.

4.2 Benchmark Test

We aim to evaluate our model's performance across the entire dataset. Employing a sliding window technique, we generated sentence chunks, each comprising 4 sentences with a stride of 2. This method yielded chunks S_i (where $i = 1$ to $N - 1$), with every 4 sentences forming one chunk and a 2-sentence overlap between adjacent windows.

To calculate $P(y|S_i, S_N)$, $P(S_N|y)$, and $P(S_N)$, we used specific models. For $P(y|S_i, S_N)$, we employed Conversational

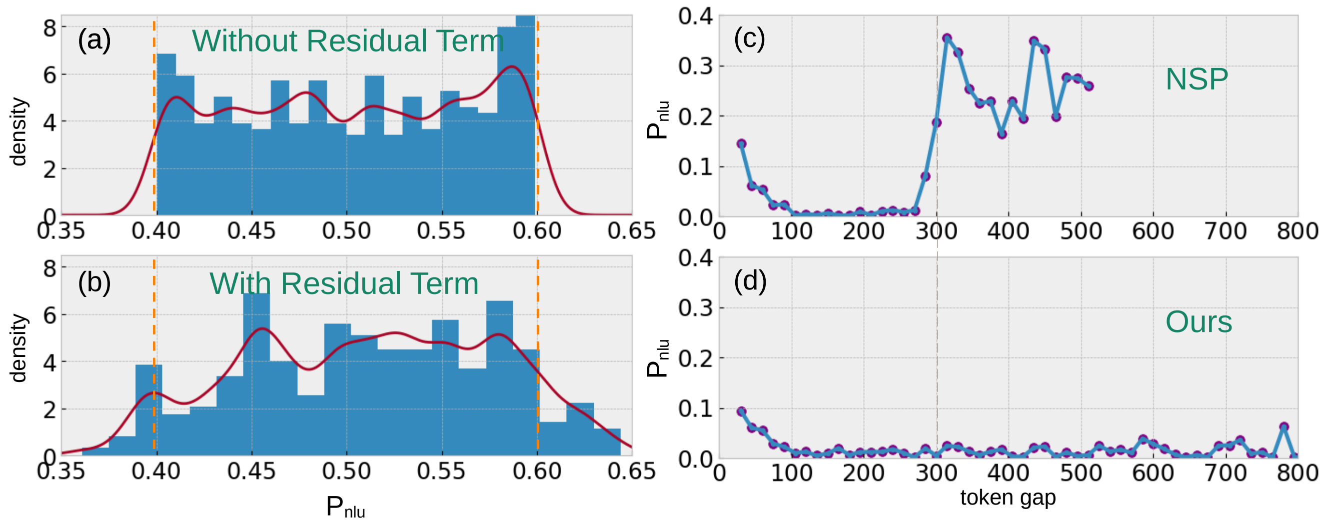

Figure 2: Impact of attention and residual terms. (a)-(b): Normalized Distribution of P_{nlu} without residual term (a) and with residual term (b) for selected uncertain examples. Red lines indicate approximate Gaussian kernel density fitting. (c)-(d): Average probability output per segmentation, categorized by token length, is shown in (c) for NSP and (d) for our model. The dashed lines denote 300 tokens. Data beyond 512 tokens were truncated in (c) due to NSP's processing limit.

BERT [24], a model trained on extensive chat data from social networks, which better captures conversational characteristics. Regarding $P(S_N|y)$ and $P(S_N)$, we randomly sampled over 100,000 sentences from conversations specific to online video streaming and from arbitrary topics, respectively. These sentences were encoded using Sentence BERT to train separate Isolation Forest models. The anomaly scores generated by these models were used to create two CDF functions for probability estimation.

Based on this setup, we observed that compared to the original BERT, using Conversational BERT significantly improved AUC performance by over 14.2%, increasing it from approximately 68.7% to around 82.9% (with accuracy from 67.8% to 80.8%) across the entire dataset. These results demonstrate that our approach performs well when faced with real-world data.

4.3 Exploration of the Residual Term

The residual term enhances NLU estimation, especially for uncertain samples when the attention term lacks confidence. To measure its effect, we select 400 examples where the attention term produces confidence levels between p_{att} = 0.4 and p_{att} = 0.6, and then measure their changes after incorporating the residual term.

The results shown in Fig. 2(a)-(b) demonstrate that the inclusion of the residual term has increased the dispersion of P_{nlu}, previously confined to the range of 0.4 to 0.6, indicating an overall boost in confidence levels. Before introducing the residual term, the model's predictions for these 400 examples resulted in precision of 0.55, recall of 0.50, and AUC of 0.47, almost resembling random guesses. However, after integrating the residual term, the metrics improved to precision of 0.62, recall of 0.65, and AUC of 0.61. This underscores the significant improvement provided by the residual term for examples that the attention term struggles to handle effectively.

4.4 Exploration of the Attention Mechanism

In contrast to using BERT directly for Next Sentence Prediction (NSP) to determine whether S_N is a reasonable context for $(S_1 + S_2 + \ldots + S_{N-1})$, our approach focuses on calculating NLU model, i.e. Eq.(1), using attention mechanisms. This approach offers advantages when handling long conversations and leap conversations. In the upcoming experiment, we aim to compare the benefits of our method with the NSP method to elucidate the role of attention mechanisms.

Token Length Dependence Here we assess the impact of token length on both models when predicting out-of-domain topic shift data. In scenarios where S_N is unrelated to the entire conversation, both models should yield results $p_{nlu} \approx 0$ (off-topic). However, segmenting conversations by token length and averaging output probabilities reveals the

Metrics	$\Delta_T \le 300$ NSP	$\Delta_T \le 300$ Ours	$300 < \Delta_T \le 512$ NSP	$300 < \Delta_T \le 512$ Ours	$\Delta_T > 512$ NSP	$\Delta_T > 512$ Ours
Precision	**0.747**	0.734	0.612	**0.697**	0.588	**0.703**
Recall	0.961	**0.983**	**0.982**	0.972	0.917	**0.980**
Accuracy	**0.818**	0.814	0.679	**0.775**	0.637	**0.783**
F1 score	0.840	**0.841**	0.754	**0.812**	0.717	**0.819**

Table 1: Comparison among different models with varying token gap lengths Δ_T. The differences between NSP and our model are minimal for narrow token gap but gradually increase as the token gap widens.

NSP model's predictions become unstable after 300 tokens (Fig.2(c)-(d)), while our model's predictions remain stable and accurate. Additionally, our model maintains performance even when token length exceeds NSP's maximum limit of 512 tokens, demonstrating the advantages of our approach.

Token Gap Dependence To further analyze attention mechanisms, we created three datasets, each containing 350 leap conversations with varying token gaps between the target sentence and the current sentence: 1) less than 300 tokens, 2) between 300 and 512 tokens, and 3) greater than 512 tokens. In each dataset, we intentionally added additional 350 topic shift conversations (half in-domain and half out-domain), turning them into binary classification tasks.

In our experiments, both the NSP and our model were used to predict outcomes on these datasets. In the third dataset, where token length exceeds the NSP model's limit, we truncated the conversation for NSP input, while our model used the entire conversation. Table 1 shows the results. NSP performs similarly to our model for small token gaps, but as the gap widens, our model outperforms NSP significantly. With token gaps surpassing 512, NSP's results become unreliable due to excluding the target sentence from its input. In contrast, our model maintains high accuracy. This experiment underscores our model's superior performance in managing conversations of varying lengths, achieving state-of-the-art results.

5 CONCLUSION

With the rapid development of large language models (LLMs), the effective utilization of LLMs in various business scenarios has become an important issue. In this paper, we propose a method that ensures user conversations with LLMs remain focused on fixed topics. This method is based on the introduction of non-linear transformations and attention mechanisms through an extension of Naive Bayes. Experimental results across various scenarios consistently demonstrate that our approach outperforms traditional methods. We believe this method will be highly beneficial for using LLMs in topic-constrained scenarios.

REFERENCES

[1] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, "A survey of large language models," *arXiv*, no. 2303.18223, 2023.

[2] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie, "A survey on evaluation of large language models," *ACM Transactions on Intelligent Systems and Technology*, p. DOI:10.1145/3641289, 2024.

[3] Y. Guo, Z. Zhang, Y. Liang, D. Zhao, and N. Duan, "Pptc benchmark: Evaluating large language models for powerpoint task completion.," *arXiv*, no. 2311.01767, 2023.

[4] P. Vaithilingam, T. Zhang, and E. Glassman, "Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models," *CHI EA '22: Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, vol. 332, pp. 1–7, 2022.

[5] S. Zhang, Z. Chen, Y. Shen, M. Ding, J. B. Tenenbaum, and C. Gan, "Planning with large language models for code generation," *arXiv*, no. 2303.05510, 2023.

[6] Y. Lin, T. Ruan, J. Liu, and H. Wang, "A survey on neural data-to-text generation," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2023.

[7] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Computing Surveys*, vol. 55, no. 248, pp. 1–38, 2023.

[8] P. P. Liang, C. Wu, L.-P. Morency, and R. Salakhutdinov, "Towards understanding and mitigating social biases in language models," *Proceedings of the 38th International Conference on Machine Learning, PMLR*, vol. 139, pp. 6565–6576, 2021.

[9] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection," *arXiv*, no. 2302.12173, 2023.

[10] E. Shayegani, M. A. A. Mamun, Y. Fu, P. Zaree, Y. Dong, and N. Abu-Ghazaleh, "Survey of vulnerabilities in large language models revealed by adversarial attacks," *arXiv*, no. 2310.10844, 2023.

[11] S.-T. Pi, C.-P. Hsieh, Q. Liu, and Y. Zhu, "Universal model in online customer service," *Companion Proceedings of the ACM Web Conference 2023*, p. 878–885, 2023.

[12] S.-T. Pi, M. Yang, and Q. Liu, "Contact complexity in customer service," *arXiv*, no. 2402.15655, 2024.

[13] S.-T. Pi, M. Yang, Y. Zhu, and Q. Liu, "Teacher-student learning on complexity in intelligent routing," *arXiv*, no. 2402.15665, 2024.

[14] S.-T. Pi, S. Srinivasan, Y. Zhu, M. Yang, and Q. Liu, "Uncovering customer issues through topological natural language analysis," *arXiv*, no. 2403.00804, 2024.

[15] A. Torfi, R. A. Shirvani, Y. Keneshloo, N. Tavaf, and E. A. Fox, “Natural language processing advancements by deep learning: A survey,” *arXiv*, no. 2003:01200, 2003.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[17] J. Devlin, M.-W. Chang, and K. T. Kenton Lee, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *Advances in neural information processing systems*, vol. 30, 2017.

[18] I. Rish, “An empirical study of the naive bayes classifier,” *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22, pp. 41–46, 2001.

[19] W. Shi and V. Demberg, “Next sentence prediction helps implicit discourse relation classification within and across domains,” *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5790–5796, 2019.

[20] Y. Sun, Y. Zheng, C. Hao, and H. Qiu, “Nsp-bert: A prompt-based few-shot learner through an original pre-training task–next sentence prediction,” *arXiv*, no. arXiv:2109.03564, 2021.

[21] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” *IEEE International Conference on Data Mining*, p. 10472172, 2008.

[22] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest based anomaly detection,” *ACM Transactions on Knowledge Discovery from Data*, vol. 6, no. 1, pp. 1–39, 2012.

[23] N. Reimers and I. Gurevych, “Sentencebert: Sentence embeddings using siamese bertnetworks .,” *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 3982–3992 (Also see, https://www.sbert.net/), 2019.

[24] DeepPavlov.ai *https://huggingface.co/DeepPavlov/bert-base-cased-conversational*, 2021.