Language Translation Model by Leveraging AI and its Impact on Banking Expansion

Prakash Parida
Sr. Consultant,
CGI Technology and Solutions Inc.

ABSTRACT

Language translation models (LTM) in the banking sector are highly revolutionary and essential for a bank's success in global expansion, especially for non-English-speaking customers. The translation should be accurate and culturally sensitive to avoid damaging the bank's reputation. Also, avoid any penalties for the bank. Expansion and outreach to diverse communities are essential in today's competitive banking market.

There are multiple advantages to adapting the language translation model in banking. The banking industry is supervised, and given the varied nature of the finance sector, it is crucial to employ language translation techniques to attract a diverse customer base, including speakers of various languages. A clear understanding of banking terms and regulations is essential for a positive customer experience. If a customer can't understand the banking terms, then there will be a lack of confidence in the process. A language translation model could make this experience easier. In many countries, banks and financial institutions are legally required to provide loan statements, disclosures, and other relevant information in the local language. Hence, accurate information must be passed to the customer. Mismanagement or misrepresentation of financial transactions can occur due to poor translation, potentially leading to economic losses, fraud, or other issues. Accurate language translation is necessary to avoid any penalties and legal consequences. A global footprint for a bank requires cross-border mergers and acquisitions. Language translation model plays a critical role in this scenario. Hence, we have discussed some of the handy cases where a language translation model can give leverage to a financial institution. We will discuss the architecture of this model and explore how AI and Machine learning can enhance its sophistication and robustness.

Keywords

Financial Transaction, Customer experience, Language Translation Model, Transformer in AI, Context Representation, Encoding, Decoding

1. INTRODUCTION

Before delving into the LTM, It needs to be understood the preference service, which enables customers to select their preferred language. It might be a secondary or primary preference.

Below is a flow diagram of the internal process for setting up preferences in the banking system.

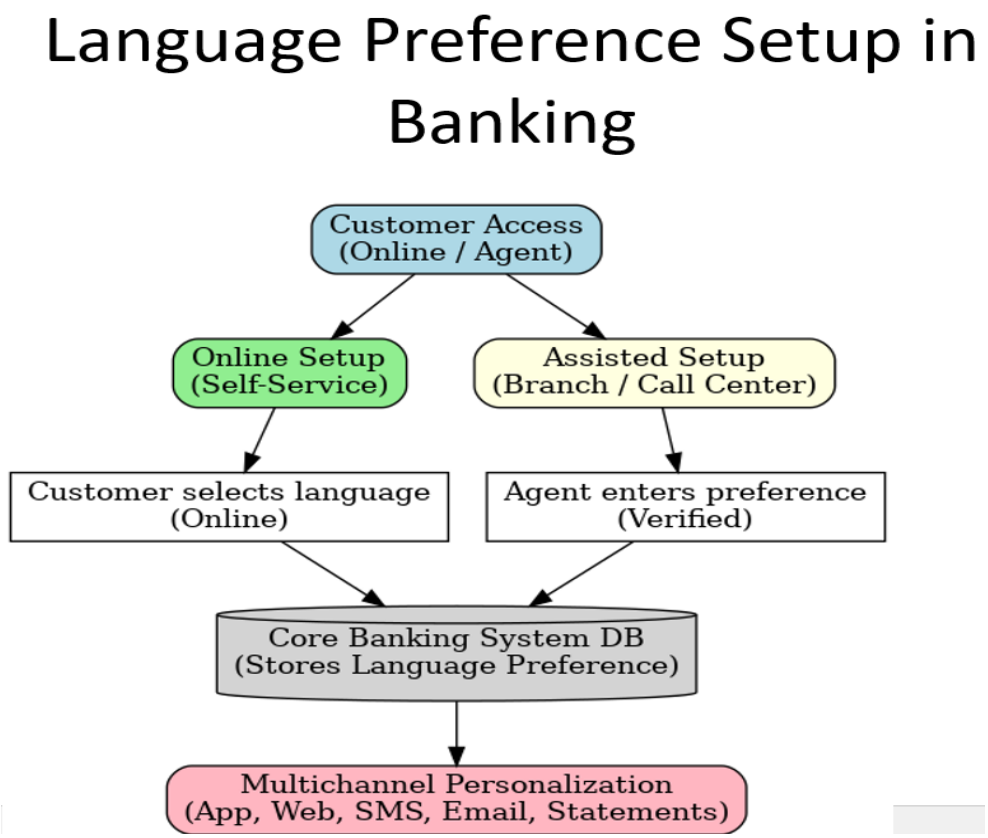

Fig. 1: Flow diagram of the internal process for setting up preferences in the banking system

In the diagram above, it can be seen that customers can either add their language preferences through the online or mobile banking system, visit a branch, or call the care center to do so. Unless there is an LTM (Language Translation Model), it is not possible to provide customers with a language preference experience.

The LTM is an AI-designed model that receives input in text or voice and accurately converts it into another language while preserving meaning and context. In banking and finance, these models must handle banking terminology, risks, compliance, and customer communication.

2. WORKFLOW OF THE LANGUAGE TRANSLATION MODEL

The workflow for AI-based LTM in banking comprises multiple steps. Below is a contextual diagram of the steps associated.

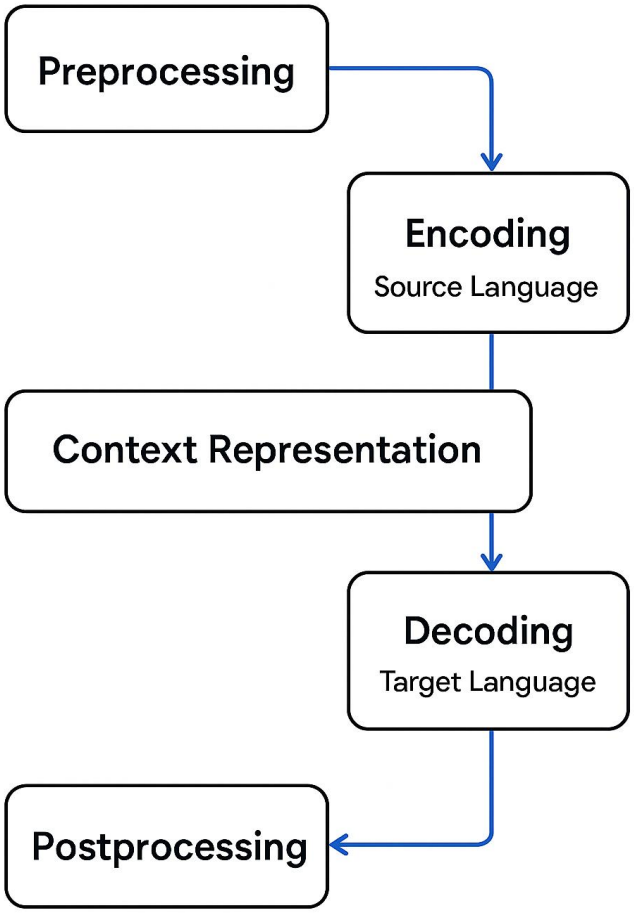

Fig. 2: High-Level Diagram for Language Translation Model

In this paper, the discussion will focus on the above steps and how AI and machine learning ease these processes.

2.1 Preprocessing of data

Data preprocessing consists of several steps, including tokenization, Normalization, and subword encoding.

Tokenization: Tokenization is an essential step in Natural Language Processing (NLP) [2]. The text is divided into small segments, known as tokens. The NLP pipeline is using these tokens. It converts the raw text into a structured format for the machine to analyze and extract context. Therefore, tokenization can transform raw text into a structured format, thereby providing a deeper understanding of its context. Various types of tokenization can be utilized for different use cases. These include word, sentence, character, and subword tokenization, among others[2].

Normalization: In data preprocessing, normalization is a data scaling technique when the data has varying ranges [3]. It improves both the model's performance and accuracy. It ensures stability in the optimization process and promotes faster convergence during the gradient-based training. It ensures that the model reaches an optimal solution more efficiently. Normalization ensures that each feature contributes proportionally to the model's learning process. However, it is not meant to delve into the normalization process in this paper. By using different normalization processes, the data is optimized and made more scalable through weighting.

Subword Encoding: In NLP data processing, subword encoding refers to breaking words into smaller units, known as subwords, for tokenization[4]. Several techniques are employed for subword encoding. One of the most common ones is Byte-pair encoding (BPE)[4]. It ensures that the most common words in a vocabulary are represented as a single token, while rare words are broken down into multiple subwords. Using this technique significantly improves vocabulary efficiency.

2.2 Encoding

After preprocessing the data through the above processes, it is input to the encoder, which converts it into a sequence of contextual embeddings. The transformer model comprises multiple stages, including positional encoding, multi-head self-attention, and Feedforward layers [5].

Positional Encoding: Positional encoding adds information about each token, generated during preprocessing. This facilitates the transformer's understanding of the relative and absolute positions of tokens, which is essential for distinguishing between words in different positions and capturing the sentence's structure [5]. Without it, the transformer would struggle to find the sequential data. The standard method for calculating positional encoding uses a sinusoidal function. The sine and cosine functions are used because they facilitate easy interpolation and generalization across sequences of different lengths. Positional encoding can be implemented using the NumPy and TensorFlow libraries. The most crucial application of positional encoding is to enhance contextual understanding, improve generalization, and mitigate symmetry issues.

Multi-head self-attention: The multi-head self-attention is a crucial component of the transformer architecture because it attends to different parts of an input sequence at a time, making it particularly effective for tasks such as machine translation and text generation. In an encoder, self-attention enables the input sequence to attend to itself. As the transformer is a critical component of the LTM, the encoder and multi-attention heads must be carefully understood. Two essential steps that need closer attention for transformer operation.

i) *Encoder self-attention*: The input sequence is injected into the input embedding and position embedding, which then outputs an encoded form for each word in the input sequence that captures the meaning and position. There are three essential parameters where the input is fed [6]. Query, key, and value in the self-attention in the encoder, which incorporates attention scores as well as the encoded representation of each word in the input sequence.

ii) *Multiple Attention Heads*: The Attention module repeats its computation multiple times (N-ways) in parallel, and each of these repetitions is called an Attention head. The attention module splits the query, key, and value in various ways and passes each split independently through a separate head [6]. All the similar attention values are combined to obtain a final attention score. Below is a pictorial representation.

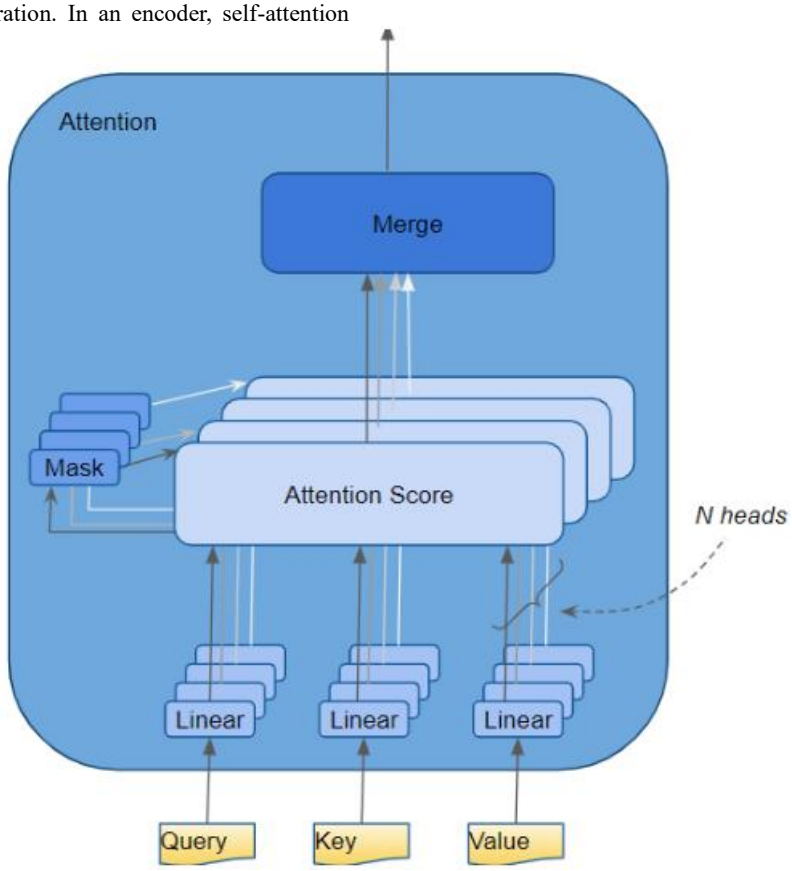

Fig 3: Multiple Attention Head Model

Feedforward Layers: The Feedforward Network (FFN) [7] is a crucial component of the Transformer architecture, enabling it to handle complex linguistic tasks with high efficiency. After the multi-head attention sublayer, the output, which maintains a dimensionality of dmodel=512, enters the FFN. It is essential to process the data sequentially, independently within each position across the sequence. The FFN, located between the encoder and decoder of the Transformer, is constructed as a fully connected layer. Each

position in the input sequence is being processed separately to maintain the position integrity.

2.3 Context Representation

The context vector receives the encoder's output. The Context Vector represents the meaning of the sentence.

There are specific core components of context generation processes, such as the self-attention mechanism. The self-attention mechanism consists of several steps: calculating Alignment Scores, applying a softmax [6]to obtain attention weights, and generating contextualized representations. In this entire process, the only words with high SoftMax scores are preserved. And lastly, the output vector is fed into a linear layer for further processing.

2.4 Output of the Encoder

The encoder layer outputs a set of vectors that represent the input sequence with appropriate context. This output will be fed into the transformer model's decoder.

The encoder's output represents contextual meaning, word positioning, and the learned representation.

The decoder uses the encoder's output. The encoder output serves as the source of the keys and value of the decoder's multi-head attention layer [8]. Additionally, the encoder output is used in the cross-attention mechanism, enabling the decoder to retrieve relevant parts of the input sequence.

2.5 Decoding

Some critical roles revolve around how the decoder performs in the transformation model. Multiple layers exist around the decoder, which will be explained in this section. The decoder has a similar structure to the encoder. It comprises two multi-headed attention layers, a pointwise feed-forward layer, and combines residual connections and layer normalization after each sub-layer.

As mentioned, the decoder has a layer similar to the encoder's, but with a slight twist. Every multi-headed attention layer has a unique mission in the decoder. The decoder process involves a linear layer, which serves as a classifier, combined with a softmax function to compute word probabilities [8]. The decoder operates in an autoregressive way. It uses a list previously generated from the outputs as its inputs, coordinating with the encoder's output.

Step 1: Output Embeddings: The input first passes through an embedding layer.

Step 2: Positional Encoding: The input passes through the positional encoding layer. The positional embeddings are then channeled into the multi-headed attention layer of the decoder, where attention scores are calculated.

Step 3: Decoder Layer Stack: The decoder consists of a stack of similar layers, each with three subcomponents.

Step 3.1: Masked Self-Attention Mechanism: It is like the self-attention mechanism, but with a significant difference. It prevents positions from attending to subsequent positions, meaning each word in the sequence is not influenced by future tokens. For example, when the attention score for the word "are" is being calculated, it is critical that "are" does not get a peek at "you", which is a subsequent word in the sequence.

Step 3.2: Encoder-Decoder Multi-head Attention: This is the second multi-head attention layer in the decoder's design [8]. In this, the encoder outputs serve as both queries and keys, while the decoder multi-head layer outputs serve as the value. This step empowers the decoder's identity and emphasizes the most relevant part of the encoder's input. After this, the output of this layer is redefined by a pointwise feedforward layer, further enhancing processing.

Step 3.3: Feed-Forward Neural Network: Like the encoder, each decoder layer includes a fully connected feed-forward network, which applies to each position identically.

Step 4: Linear Classifier: The data's journey through the transformer model culminates in its passage through the final linear layer, which serves as a classifier. If there are 500 distinct classes representing 500 different words, the classifier output would be an array of 500 elements. This output is then fed into a softmax layer, which normalizes it to the range 0 to 1. The highest probability score is the key; its corresponding index directly points to the predicted word.

Step 5: Normalization and residual connection: Each sublayer (masked self-attention, encoder-decoder attention, feed-forward network) is followed by a normalization step and a residual connection.

Output of the Decoder: The final layer is transformed into a predicted sequence through a linear layer followed by a softmax to generate probabilities over the vocabulary [8]. The decoder incorporates the outputs into the growing list of inputs and then proceeds with the decoding process. This repeats until the model predicts the specific token. It's important to note that the decoder is not limited to a single layer. It can be structured with N layers.

There are multiple language transformer models available. Examples are: BERT, LaMDA, ChatGPT, etc.

2.6 PostProcessing

Multiple postprocessing methods are being used at the end of the translation process. Some of them are mentioned below:

De-tokenization: It removes spaces between tokens when needed [9].

One example could be as follows:

"I ' m hungry .", the output should be: "I'm hungry."

Truecasing: Identifies the proper capitalization. It ensures that the appropriate capital word is in the proper place.

Example:

"the president of the usa" , the output should be: "The president of the USA"

Punctuation Normalization:

It ensures correct usage of the punctuation.

Example:

"Hello , Sir !", should be "Hello, Sir!"

Grammar Correction: There should be grammar rules to fix if there are any issues.

Other methods are being used, such as format restoration, language adjustment, filtering offensive language, and quality estimation.

3. IMPLEMENTATION CHALLENGES

In banking, there are many challenges when implementing language translation. The accuracy of language translation

matters most when dealing with any statements, notices, etc. that the Bank produces for customers. If there are any issues, not only does it damage the financial institution's reputation, but it can also result in a hefty penalty.

There are three qualities of a good translation [10]: accuracy, fluency, and cultural sensitivity.

- Accuracy: A good translation should accurately convey the message, meaning, and intent of the text. It should be free from errors and mistranslations, and ensure that the content is reliable and credible.
- Fluency: Fluency signifies the translation's natural flow [10]. It should be read in the same way it was originally crafted, maintaining the same engagement and flow.
- Cultural Sensitivity: Cultural Sensitivity demonstrates the translation to the cultural norms, references to the target audience. It ensures that the cultural understanding is intact.

However, there are methods for checking translation quality. Some of the methods are provided below:

1. Does the Translator follow the source text?
    - Understanding the source text is essential. Without understanding the source text, it is not possible to grasp the actual meaning.
    - Misunderstanding of the text can lead to mistranslation. It can cause miscommunication and confusion.
2. Is the translation localized for the intended audience?
    - Localization is the process of adapting the content so that it suits cultural, linguistic, and social preferences for the intended audiences.
    - Failure to do a proper localization can cause cultural insensitivity and offence, which can cause significant damage to the financial institution.
3. Are the names, trademarks etc preserved from the source text?
    - Preserving the brand names, trademarks, and cultural references is essential for maintaining quality and consistency.
    - Failure to do so can lead to mistrust, confusion, and even legal disputes.
4. Are the numbers and measurements translated accurately?
    - The precision in translating the numbers, units, and measurements is crucial. Mistakes in these areas can cause safety risks.

By leveraging AI translators, the above is taken care of, but there should still be native-language speakers who can verify and make the best judgment on the translation. And if any modifications are needed, they should be made appropriately.

4. CONCLUSION

In this research paper, the core method of language translation has been discussed and why it is essential for banks to adopt it. Banks want to provide ultimate facilities to clients, including translation into their native languages as a preferred service. It is challenging, but it can foster positive sentiment among native-language speakers. Localization, cultural sensitivity, and accuracy are the challenges; however, with the adoption of AI, much has been streamlined, and translation accuracy has been achieved. The continuous feed of training data to the AI models is quite significant for achieving accuracy and localization. There are cloud-based AI tools that can handle simple translations. However, for many financial institutions, it has now become necessary to adopt the translation process. Companies such as TransPerfect and Lionbridge are doing a phenomenal job in this space. But it is still essential that native speakers(partial human intervention) verify translations to ensure they are perfect.

5. REFERENCES

[1] (2024) Eli Gutialban : Understanding Natural Language Processing Basics.

[2] (2024) Erhan Arslan: Natural language Processing: Deep Dive into Text Preprocessing and Tokenization— Step 2

[3] (2024) Sejal Jaiswal: What is Normalization in Machine Learning? A Comprehensive Guide to Data Rescaling

[4] (2021) Chetana Khanna: Byte-Pair Encoding: Subword-based tokenization algorithm

[5] (2025) Positional Encoding in Transformers (geeksforgeeks)

[6] (2021) Ketan Doshi: Transformers Explained Visually (Part 3): Multi-head Attention, deep dive

[7] (2024) Sandaruwan Herath: The Feedforward Network (FFN) in The Transformer Model

[8] (2024) Josep Ferrer: How Transformers Work: A Detailed Exploration of Transformer Architecture

[9] (2016) Detokenization in Machine Translation (peteris.rocks)

[10] (2025) Liraz Postan: How To Evaluate Translation Quality: 8 Point Checklist