Contextual Modeling for Document-level ASR Error Correction

**Jin Jiang[1,2], Xunjian Yin[1], Xiaojun Wan[1], Wei Peng[3], Rongjun Li[3]
Jingyuan Yang[3], Yanquan Zhou[2]**
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Beijing University of Posts and Telecommunications
[3]Artificial Intelligence Application Research Center, Huawei Technologies
jiangjin@bupt.edu.cn, wanxiaojun@pku.edu.cn

Abstract

Contextual information, including the sentences in the same document and in other documents of the dataset, plays a crucial role in improving the accuracy of document-level ASR Error Correction (AEC), while most previous works ignore this. In this paper, we propose a context-aware method that utilizes a k-Nearest Neighbors (kNN) approach to enhance the AEC model by retrieving a datastore containing contextual information. We conduct experiments on two English and two Chinese datasets, and the results demonstrate that our proposed model can effectively utilize contextual information to improve document-level AEC. Furthermore, the context information from the whole dataset provides even better results.

Keywords: Document-level, ASR Error Correction, k-Nearest Neighbors

1. Introduction

ASR Error Correction (AEC) is a task that aims to rectify errors in a text produced by automatic speech recognition (ASR) systems. Document-level ASR error correction (Doc-AEC) is a specialized form of AEC that focuses on correcting errors in a document, where the sentences within the document have contextual relationships. Unfortunately, most previous ASR error correction methods primarily focus on sentence-level correction (Zhao et al., 2021; Dutta et al., 2022), often neglecting contextual information.

However, in many ASR scenarios, such as news reports, telephone conversations, and audiobooks, contextual information plays a important role in accurately recognizing speech. An example of the news reports text is shown in Table 1, in order to correct the subject "他(He)", it is necessary to have contextual information such as "朱婷(Zhu Ting)". To solve such a problem, we try to model the document context in the document-level AEC. Nevertheless, there are limited studies (Wang et al., 2021, 2022b) on incorporating contextual information into Doc-AEC.

Document-level text generation tasks share a similar task paradigm and the approaches used to model documents are generally applicable. Meanwhile, there has been advancement in machine translation at the document level, where techniques such as including an additional encoder for contextual information for each sentence (sent2sent) (Zheng et al., 2020; Khandelwal et al., 2020b) and directly inputting multiple sentences from the entire document (doc2doc) (Wang et al., 2021, 2022b) have been developed. These two methods construct the connection of document context by expanding the scope of attention. Nevertheless, this approach encounters two issues: the length of document context is constrained by self-attention computation and incorporating too much unselected context brings additional noise. Moreover, it has been observed that the methodology relying on nearest neighbor retrieval has made advancements in numerous tasks (Khandelwal et al., 2020b; Wu et al., 2021) and may be utilized as a technique for context modeling. Inspired by the above works, for document-level AEC, we propose a kNN-based context-aware model, which augments the model via kNN retrieval from a context datastore. Our model can effectively solve the aforementioned issues.

Context	本场比赛朱婷三七次扣球得到二十一分。
Trans.	Zhu Ting scored twenty-one points in this game with three seven buckets.
Label	她(She)还凭借拦网和发球分别拿到七分和一分。
Trans.	She also had seven points and one point with her block and serve, respectively.
BART	他(He)还凭借拦网和发球分别拿到七分和一分。

Table 1: Example of document-level AEC errors, including context, label and BART output.

Our context-aware model retrieves contextual information from a datastore to acquire useful information for rectifying the current sentence. Therefore, the construction of the datastore is crucial and determines which contexts we model. For document-level tasks, the document itself is the most important context we need to model. Therefore, we can build a context datastore based on the

20-25 May, 2024. © 2024 ELRA Language Resource Association: CC BY-NC 4.0

context within a document, which we call the doucment context; however, the context information is not only limited to the current document. Considering that other documents in the dataset have the same domain as the current document, we extend the context from the current document to all documents in the dataset. At this point, the datastore saves contextual information for the entire dataset, which we call the dataset context. From document context to dataset context, our model can retrieve from richer contextual datastore. In addition, considering the real-time needs of speech recognition, we often need online models. The online contexaware AEC model needs to correct errors and build the datastore synchronously in real time. In this online case, the content of the datastore contains only the preceding part of the current sentence and can not model the following sentences. The offline model, as opposed to the online model, can model both the preceding and following sentences together. In a real ASR scenario, online and offline models can co-exist and complement each other. The online model first corrects errors in real time and builds a datastore. When the online error correction is completed, the datastore of the online model can be used as the initialization of the offline datastore. In general, we improve the construction of the datastore in two dimensions: timing of data construction（online and offline) and scope of the datastore context(document context to dataset context).

Finally, we develop a document-level ASR error correction model with a kNN attention layer that utilizes k-Nearest Neighbors to model context information from the same document and the whole dataset, for both online and offline models. The main contributions of this article are as follows:

- We propose a new context modeling approach for document-level ASR error correction, which incorporates contextual information using a kNN retrieval mechanism.
- We extend the online model to the offline model and broaden the context scope from document to entire dataset, thereby maximizing the contextual information that can be modeled.
- Experiments[1] on four document-level AEC datasets demonstrate that our model effectively utilizes contextual information to enhance the performance of the error correction model.

[1] https://github.com/jiangjin1999/context_ASR

2. Background

2.1. Context Definition

In natural language processing, the term "context" has a rich meaning, referring to the information and surrounding context related to a piece of text. Depending on the source of the information, context can be divided into the following four types : the current sentence, the k sentences before and after the current sentence, the document in which the current sentence is located, and the dataset in which the current document is located. In this article, we are modeling the context of the entire document in which the current sentence is located and the entire dataset in which the current document is located.

2.2. Transformer: Local Attention

The Transformer model consists of standard attention layers, including "Scaled Dot-Product Attention" and "Multi-Head Attention". In particular, in this paper, Q, K, V refer to the matrices after linear projections. The corresponding equations are as follows.

Scaled Dot-Product Attention:

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

Multi-Head Attention:

$$\begin{aligned} &\text{MultiHead}(Q, K, V) \\ &\quad = \text{Concat}(\text{head}_1, \ldots, \text{head}_\text{h})W^O \\ &\text{where } \text{head}_\text{i} \\ &\quad = \text{Attention}(Q_i, K_i, V_i) \end{aligned} \quad (2)$$

Where $Q_i, K_i, V_i \in \mathbb{R}^{bs \times l \times d_{head}}$ are the i-th head matrices after linear projections. Variable definition: bs: batch size, h: head num, l: sequence length, d_{model}: dimension of each head.

2.3. Nearest Neighbor-based Model

k-nearest neightbors (kNN) algorithm has wide applications in the field of natural language processing. Its main goal is to improve the model performance by retrieving the top-k results based on distance measurement in a datastore. The datastore format includes a series of key-value pairs$((\mathcal{K}, \mathcal{V}))$, where the values are returned by searching the keys.

One popular model is kNN-MT (Khandelwal et al., 2020b). During the training phase, this approach records the decoder representations $(f(c_i))$ for each input (w_i) in the training data $(\mathcal{D})$ as keys and their corresponding target tokens as values:$(\mathcal{K}, \mathcal{V}) = \{(f(c_i), w_i) \mid (c_i, w_i) \in \mathcal{D}\}$.

Furthermore, the kNN algorithm is not limited to this. The key used for searching and the key value

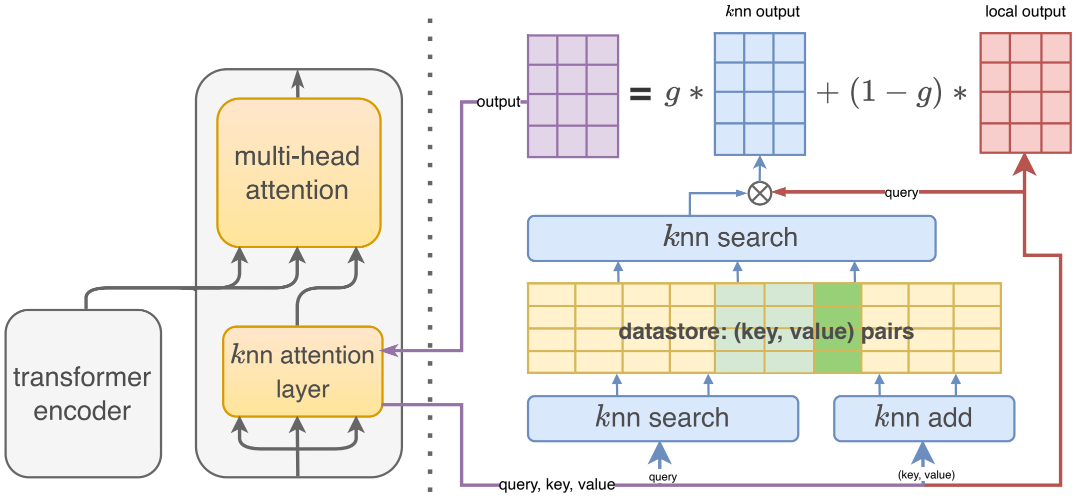

Figure 1: An overview of our context-aware AEC model. The left half of the figure is the simplified end-to-end model schematic. The right half shows the computation of the kNN-augmented attention layer. The tensor flow starts from the purple line, then divides into local attention and kNN attention, and finally the weighted additions to output.

used for storage can also take other forms. In this paper, the key used for searching is the query in self-attention and the key-value used for datastore is the key-value in self-attention.

3. Methodology

The core of our work lies in enhancing the seq2seq model with a kNN attention layer, enabling the model to access a datastore storing contextual information. In section 3.1, we will present the overall structure of the model, as well as the training and test processes when incorporating the kNN attention layer. Section 3.2 will elaborate on the computational details of the kNN attention layer. Section 3.3 will outline how we batch processing document data in the document-level scenarios. In section 3.4, we will discuss how we build different datastores for online and offline models, which determine the contextual information we use.

3.1. Model Structure

As shown in Figure 1, our context-aware AEC model is an encoder-decoder architecture. The encoder is a vanilla transformer encoder while the decoder is a transformer decoder that incorporates a kNN attention layer. The model takes uncorrected ASR results as input and generates corrected results as output.

The kNN attention layer introduces external datastore through a nearest neighbor retrieval mechanism. Specifically, datastores are constructed for the train, test and validation stages respectively. During the training phase, the model updates its parameters based on the retrieval results from the datastores. In the inference phase, the model utilizes retrieval results to assist in error correction without updating parameters.

3.2. kNN-augmented Attention Layer

In our model, one of the layers in the transformer decoder is a kNN-augmented attention layer. Unlike the standard decoder layer, it performs an approximate k-nearest neighbor search operation in an external datastore. The computation of this layer consists of two parts: local attention and kNN attention. The right side of Figure 1 explains the tensor flow of these two parts in the model. The red flow represents the local attention, which is computed in the same way as the standard self-attention calculation, as we have introduced in Section 2.2. The blue flow and datastore refer to the kNN attention computation.

The computation of kNN attention involves both **kNN search** and **attention calculation**. For kNN search, the input is queries that are the same as local attention. After similarity matching, it returns the top-k key-value pairs for each token in each query, as shown in Equation 3.

$$knn\_K, knn\_V = k\textbf{NN search}(Q) \tag{3}$$

For attention calculation based on kNN search results, we first compute the dot product between each query and the retrieved keys to obtain the attention matrix. Then, we calculate the weighted sum of the attention matrix and the retrieved values. Unlike local attention, each query corresponds to k

key-value pairs, as shown in the following formula:

$$\begin{aligned}
&k\text{NN\_MultiHead}(Q, knn\_K, knn\_V) \\
&\quad = \text{Average}[\text{knn\_heads}, K]W^O \\
&\text{where } \text{knn\_heads} \\
&\quad = \text{Concat}\left(\text{knn\_head}_1, \ldots, \text{knn\_head}_\text{h}\right) \\
&\text{where } \text{knn\_head}_\text{i} \\
&\quad = \text{Attention}\left(Q_i, knn\_K_i, knn\_V_i\right)
\end{aligned} \tag{4}$$

Where $knn\_K_i, knn\_V_i \in \mathbb{R}^{bs \times l \times d_{head} \times k}$ are the i-th head retrieved matrices, $\text{knn\_heads} \in \mathbb{R}^{bs \times h \times l \times d_{head} \times k}$ and k represents retrieving neighbors number.

By computing local attention, we can obtain the local output (red matrix in Figure 1). By computing kNN attention, we can obtain the kNN output (blue matrix in Figure 1). The final output(purple matrix in Figure 1) of kNN-augmented attention layer is obtained by applying a weighted calculation using the gate hyper-parameter g .

3.3. Data Batching

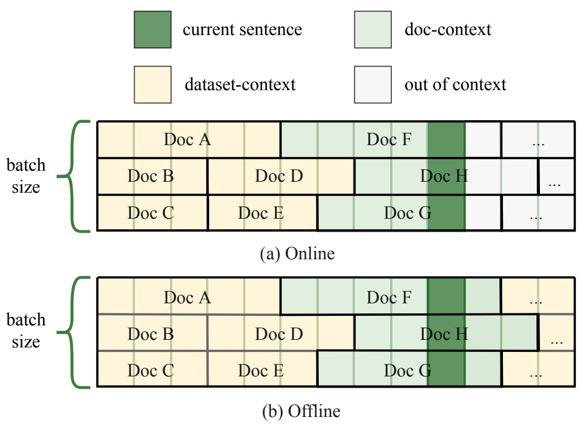

Figure 2: An illustration of data batching and context datastore construction strategies for online and offline. Legend implication: "current sentence: current correction sentence, doc-context: other sentences of the document where the current sentence is located, dataset-context: other sentences of the dataset where the current sentence is located (e.g., the whole test set), out of context: sentences that have not been processed by in online mode."

Figure 2 illustrates how we batch process documents of different lengths in a document-level scenario. To process multiple documents in parallel, we set the batch size to the first dimension of the datastore, which corresponds to the rows in the grid shown in the figure. Then, we sort the documents based on the rules of top-to-bottom (e.g., Doc D on the second row and Doc E on the third row) and less-to-many (e.g., Doc G on the third row and Doc H on the second row). Finally, we align and pad the documents to the length of the longest row.

In addition, in our experiments, the batch sizes for train, test and validation are often different. For example, in the AISHELL-1 dataset, there are 340/20/40 documents in the train/test/validation sets. Due to GPU memory constraints, the maximum batch size for this dataset is 64, while there are only 20 and 40 documents in the test and validation sets. Therefore, in order to process the different documents in parallel, we can only set the final batch size to 64/20/40.

3.4. Context Datastore Construction

As shown in Figure 2, the construction of the context datastore can be divided into two modes: online and offline. Online means that the datastore is constructed using only the preceding text of the current sentence, while offline means that the datastore is constructed using the preceding and following text of the current sentence. For a document-level task, each mode can be further categorized into two types: doc-context and dataset-context. "doc-context" refers to the use of sentences in the document as contextual information to construct datastore, while "dataset context" refers to the use of all documents in the current dataset as contextual information to construct datastore. In general, our model has four different types of context datastore construction. Different contextual datastore construction types correspond to different contextual information.

Online The online model initializes a datastore at the beginning and updates it while correcting the input sentence. During the correction, the model stores the key-value pairs of each token in the current sentence to the datastore through kNN add operation k**NN add**(K, V). Therefore, when correcting in the i-th sentence, the online model can only search within the datastore comprised of the preceding sentences (the 1st to the $(i-1)$th sentences). That is, the datastore of the online mode includes only the preceding text of the current sentence. For doc-context, the preceding sentences only include the sentences from the same document that comes before the current sentence. For dataset-context, the preceding sentences include all sentences in the document that have been stored.

Offline The offline model constructs a datastore in advance. In practical scenarios, we can first obtain real-time correction results through the online model and then use the datastore stored in the online model as the initialization of the offline model datastore. This way, the offline model can simultaneously obtain information from both the preceding and following sentences for the current sentence. Therefore, in offline mode, the doc-context model can search the datastore composed of the preceding and following sentences in the current document. The dataset-context model can search all documents in the current dataset.

4. Experiments

4.1. Datasets and Evaluation Metrics

Considering that there are no publicly available document-level ASR Error Correction datasets,we construct two Mandarin datasets and two English datasets from ASR datasets containing document formats: AISHELL-1 (Bu et al., 2017), HKUST (Pascale Fung, Shudong Huang, David Graff, 2005), LIBRISPEECH_CLEAN and LIBRISPEECH_OTHER (Panayotov et al., 2015). Details about the datasets, including data statistics and the ASR model used, can be found in Appendix A.1.

For evaluation metrics, we use character error rate (CER) and character error rate reduction (CERR), word error rate (WER) and word error rate reduction (WERR) as metrics for Chinese and English respectively.

4.2. Training and Inference Details

For the training phase, first, we use the pre-trained BART to train a sentence-level AEC model, which serves as our baseline model. Subsequently, we use the baseline model as the initialization of our Context-aware AEC model. Then, our model retrieves the context datastore through the kNN attention layer and incorporates the retrieval results into the model, and finally the model parameters are updated. For the inference phase, the model does not update the parameters. Inference is performed directly by incorporating the retrieval results through the kNN attention layer. In addition, we construct separate datastores for the training and inference phases. For the online model, we build the datastore in real time during the training and inference phases. For the offline model, we directly use the datastore of the online model as the initialization of the datastore of the offline model.

In the kNN attention layer, we employ approximate kNN search to speed the model through FAISS library (Johnson et al., 2019). The hyperparameters and model configurations are shown in Appendix A.2

4.3. Baseline Models

Sentence-level ASR error correction based on pre-trained **BART** is our first baseline. Compared with this baseline, it is possible to verify whether our approach can effectively use contextual information to improve error correction performance. Since there is no model on document-level ASR error correction, we construct two baseline models based on two paradigms for document-level text generation tasks. For sent2sent, we construct a **dual encoder** model to separately encode the current sentence and the sentences before and after the current sentence to introduce contextual information. For the document-to-document model, we use a **sliding** window to input the preceding and following k sentences of the current sentence as one document into the error correction model.

4.4. Main Results

The experimental results are listed in Table 2. The first four rows of the table are the uncorrected results and the results of the three baselines. Comparing the three baseline models, we can see that the two document-level baseline models (dual encoder and sliding k) perform even worse than the sentence-level BART baseline in most cases. Only the dual encoder baseline model achieves some improvement in HKUST and the two English datasets. This indicates on the one hand that the context in ASR error correction is of limited help to the error correction task, and on the other hand that directly introducing the sentences before and after the current sentence may introduce additional noise that makes the model less effective.

The last four rows of the table show the results of our proposed model for four different datastore construction methods. The four different construction methods imply different contextual information. The specific differences are presented in section 3.4. Compared with the baseline, our four models achieve better results in most cases. This shows that our approach can effectively use context to improve error correction. The comparison between the four datastore construction methods shows that the offline models all work better than the online models, which indicates the significance to introduce information about both the preceding and the following context of the current sentence. In addition, the models using the dataset context to build the datastore show a small improvement over the models using only the document context, suggesting that documents other than the current one can also help the model to correct errors better. Overall, the offline dataset context model achieves the best results on all four datasets, which demonstrates that using more contexts in our model can lead to better results.

Method	AISHELL-1		HKUST		LIB_CLEAN		LIB_OTHER	
	CER↓	CERR↑	CER↓	CERR↑	WER↓	WERR↑	WER↓	WERR↑
RAW	4.312	-	27.60	-	4.553	-	9.136	-
BART	3.528	18.18	21.36	22.61	1.967	56.80	4.318	52.74
Dual (sent2sent)	3.569	17.23	21.10	23.55	1.958	57.00	4.304	52.89
Sliding k(doc2doc)	3.638	15.63	22.84	17.25	1.995	56.18	4.855	46.86
Online AEC(doc)	3.412	20.87	21.23	23.08	1.844	59.50	4.252	53.46
Online AEC(dataset)	3.379	21.64	20.68	25.07	1.835	59.70	4.245	53.54
Offline AEC(doc)	3.407	20.99	20.96	24.06	1.827	59.87	4.237	53.62
Offline AEC(dataset)	**3.336***	**22.63***	**20.66***	**25.14 ***	**1.822***	**59.98 ***	**4.237***	**53.62***

Table 2: The performances of our kNN Contex-aware AEC model in four different datastore construction methods and baseline models. (doc) and (dataset): document contex and dataset context. Dual: Dual Encoder baseline. LIB_CLEAN and LIB_OTHER: LIBRISPEECH_CLEAN and LIBRISPEECH_OTHER datasets. "*" indicates statistically significant at $p < 0.05$ compared to the BART baselines.

5. Analysis and Discussion

5.1. Error Type Statistics

The percentage of errors that lack context illustrates the importance of document ASR error correction. Taking the AISHELL-1 dataset as an example, we perform statistics on the results after applying the baseline model for error correction. The statistical results are shown in table 3. The proportion of these errors due to lack of context is nearly one quarter. In addition, previous work on AEC at the sentence level performs relevant statistics. Table 5 in Zhao et al. (2021) statisticizes the types of errors in the dataset they construct. The types of errors with insufficient context account for 37% of all errors.

Error type	Percentage
Insufficient Context	**24%**
Nouns Error	34%
Equal Expression	10%
Label Error	7%
Model Error	25%

Table 3: Error type statistics. "Insufficient Context" is an error caused by lack of contextual information. "Nouns Error" is a noun or entity error. "Equal Expression" is when the output of the model does not match the label but is actually correct. "Label Error" is when the output of the model is correct but the label is erroneous. "Model Error" is an error that cannot be correctly corrected by the current model, and may be related to features such as pronunciation.

Context	李斯达(name)表示自己跟周云露并没有深仇大恨
Label	目前李斯达(name)被关押在朝阳看守所
BART	目前李思达(name)被关押在朝阳看守所
Ours	目前李斯达(name)被关押在朝阳看守所
Context Trans.	深入贯彻落实科学发展观。 Deeply implement the scientific concept of development.
Label Trans.	编者按：为深入贯彻落实中央八项规定精神。 Editor's note: In order to thoroughly implement the spirit of the eight central regulations.
BART	编者按：为深入管撤落实中央八项规定精神。
Ours	编者按：为深入贯彻落实中央八项规定精神。

Table 4: Some examples from AISHELL-1. The words in green mean context information and the words in red mean error and the words in blue mean label.

5.2. Case Study

From the experimental cases, we can see that contextual information help the model correct some errors that cannot be corrected by sentence-level baselines. As shown in Table 4, our context-aware AEC model corrects 李思达 to 李斯达 by retrieving the correct person name that appears in the document context. For the second example, its context "深入贯彻落实科学发展观。" is a sentence from another document in the dataset. The fact that the model can correct "管撤" to "贯彻" indicates that the model further improves the results by retrieving over a larger context datastore. More examples can be seen in Appendix A.3.

5.3. Ablation Study

Inspired by Li et al. (2020), we wonder if context brings better correction performance or just noise brings help. Therefore we perform ablation study as shown in Table 5. We can see that either "shuffle" or "random" gives worse model results. Therefore, we can conclude that document context does help.

Context	AISHELL-1	HKUST
Contex AEC	3.407	20.96
→ shuffle	3.436	21.13
→ random	3.501	21.50
Contex	**LIB_CLEAN**	**LIB_OTHER**
Contex AEC	1.827	4.237
→ shuffle	1.983	4.294
→ random	1.942	4.466

Table 5: Ablation test results with Offline doc-contex AEC model. Metrics are CER and WER. "shuffle" means to break up the document data. "random" means to replace the retrieved results with a random vector.

5.4. Information Retrieved from Context

Table 6 shows what information the model retrieves in the document context. For example, the input: "目前李" prefers to output "斯" instead of "思" because it retrieves "李斯达 " in the context.

5.5. Effect of Key Hyperparameters

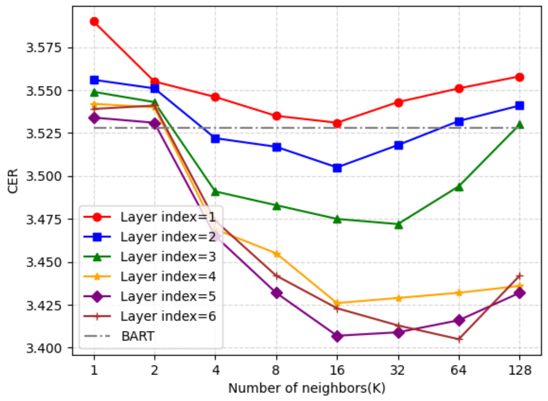

Figure 3: Effect of the number of neighbors retrieved and kNN attention layer index on AISHELL-1 datasete. The performance of the baseline is marked with a gray dashed line. The Y-axis is the CER metric, and the smaller this metric is the better the model is.

Number of neighbors We study the effect of the number of nearest neighbors (k) the model retrieved from the context datastore. Figure 3 shows that performance improves rapidly as the number of neighbors increases, but begins to decline gradually once it reaches around 16, 32, and 64. Therefore, we only need to retrieve 16 neighbors and we can obtain a comparable results with 32 or 64 neighbors.

kNN layer index We experiment with adding the kNN attention layer to all layers in the transformer decoder. As shown in Figure 3, the performance of the model improves when incorporating the kNN attention layer into a deeper layer of the model, like layer 4, layer 5, or layer 6. The optimal performance is achieved when the attention layer is added to the 5-th layer.

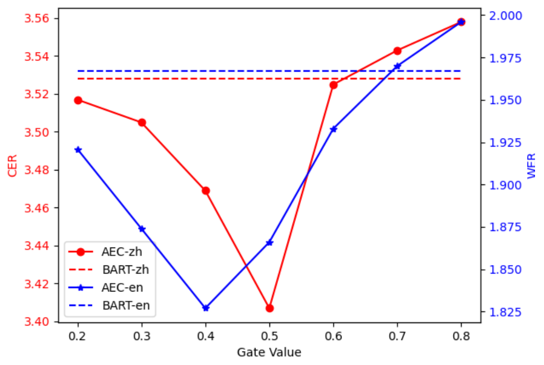

Figure 4: Effect of the interpolation parameter gate value. AEC-zh and BART-zh represent the results of our contextual AEC model and BART baseline on the Chinese dataset ASHELL-1. AEC-en and BART-en represent the results on the English dataset LIBRISPEECH_CLEAN. The left Y-axis and the right Y-axis indicate the metric results in Chinese and English, respectively.

Gate Value As shown in Figure 4, the best results of the model are obtained for gate values around 0.4 and 0.5. In addition, smaller gate values will weaken the contextual information but not worse than the baseline. However, larger gate values may result in very poor results by weakening the original information of the current sentence.

5.6. Inference Time

As shown in Table 7, we investigated the inference time of the context-aware AEC model. It can be seen that there is not much difference between the inference speed of our model and the baseline. This is partly because we use the approximate k-nearest neighbor retrieval algorithm and

Query Index	Input	Output	Uncorrected sentence	Retrieved Index	Retrieved context
13004	目前李	斯	目前李思达(name)...	12922	李斯达(name)
3964	为深入	贯	为深入管撤落实...	4203	深入贯彻落实

Table 6: Examples of document context retrieval in case study. The "Query Index" represents the position of the current token in the document in the "Input", while the "Retrieved Index" represents the position of the retrieved token in the document context. The "Uncorrected sentence" is the output of the baseline model, while the "Output" is the result of the model after incorporating the "Retrieved context"

ms/sent	k	doc-context	dataset-context
BART	-	85.42	
Contex AEC	4	89.70($\times 1.05$)	109.3($\times 1.28$)
	8	95.67($\times 1.12$)	122.1($\times 1.43$)
	16	101.6($\times 1.19$)	128.1($\times 1.50$)
	32	104.2($\times 1.22$)	132.4($\times 1.55$)

Table 7: Inference time of our context-aware AEC model with doc-context and dataset-context and BART baseline. All results are tested on 112 cores Intel(R) Xeon(R) Gold 6330 CPU 2.00GHz with a A40-48GB GPU

the FAISS index supports GPU computation. On the other hand, the size of our datastore is small. Compared to the kNN-MT method which uses the entire training set to construct the datastore, our largest datastore only includes the test data itself.

5.7. Evaluation on ChatGPT

dataset	model	CER
AISHELL-1	BART	4.32
	Context AEC	**3.37**
	Chatgpt-sent	5.89
	Chatgpt-doc	6.48

Table 8: Chatgpt evaluation results on AISHELL-1. Chatgpt-sent inputs single sentence and Chatgpt-doc inputs entire document.

The emergence of ChatGPT has had a significant impact on natural language processing tasks. To evaluate the effectiveness of our method, we conduct an experiment to compare ChatGPT's performance to that of the BART baseline and our model in the ASR error correction task. The results, presented in Table 8, indicate that ChatGPT performs even worse than the BART baseline on the AISHELL-1 dataset. Further details, including the model and prompts used, can be found in Appendix A.4.

6. Related Work

Document-level Error Correction and Machine Translation There is limited research addressing document-level tasks for error correction, including document-level AEC. Yuan and Bryant (2021) proposed a document-level correction model and metrics for grammatical error correction tasks. For ASR error correction, no research has been devoted to addressing document-level AEC tasks. However, Wang et al. (2021, 2022b) introduced information from the context list by adding an additional context encoder to the original ASR error correction model. Bekal et al. (2021) solved entity-related problems in AEC tasks by utilizing kNN search of an entity datastore.

In Machine translation, document-level machine translation has received increasing research attention (Voita et al., 2018; Zheng et al., 2020; Yang et al., 2019; Zhang et al., 2022; Feng et al., 2022; Sun et al., 2022). Some studies (sent2sent) (Werlen et al., 2018; Zhang et al., 2018; Guo and Le Nguyen, 2020; Zheng et al., 2020) have introduced additional context encoders on top of sentence-level machine translation and are jointly trained with the original model. Other studies (doc2doc) (Ma et al., 2020; Bao et al., 2021) concatenate multiple sentences or directly output the entire document for translation.

Retrieval Augmentation Retrieval-Augmented Paradigm has been broadly utilized in a variety of tasks such as language modeling (Guu et al., 2020) and machine translation (Gu et al., 2018). Several researchers have used external datastores to enhance language modeling performance, including Khandelwal et al. (2020a); Borgeaud et al. (2022); Wu et al. (2021); Zhong et al. (2022). For text generation tasks, Khandelwal et al. (2020b) have created data stores from training sets and utilized a nearest neighbor classifier over a vast datastore to enable machine translation. Moreover, comparable strategies are broadly utilized in other MT-like text generation tasks (Zheng et al., 2021; Wang et al., 2022a; Yin et al., 2022).

7. Conclusion and Future Works

To summarize, we propose a context-aware model for document-level ASR error correction that utilizes k nearest neighbors to model document and dataset context. Our model effectively utilizes contextual information to enhance the performance of the error correction model, demonstrating the importance of incorporating contextual information in ASR scenarios. For future work, we are interested in extending our kNN-based context-aware AEC architecture to other text generation tasks like document-level machine translation.

8. Limitations

Our proposed context-aware AEC model is a novel and general model. The model can be applied to all document-level text generation tasks. Therefore, in order to validate the generality of the model, we should experiment on other tasks such as document-level machine translation.

9. Acknowledgements

We appreciate the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

10. Bibliographical References

Guangsheng Bao, Yue Zhang, Zhiyang Teng, Boxing Chen, and Weihua Luo. 2021. G-transformer for document-level machine translation. In *ACL/IJCNLP*.

Dhanush Bekal, Ashish Shenoy, Monica Sunkara, Sravan Bodapati, and Katrin Kirchhoff. 2021. Remember the context! asr slot error correction through memorization. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 236–243. IEEE.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Samrat Dutta, Shreyansh Jain, Ayush Maheshwari, Ganesh Ramakrishnan, and Preethi Jyothi. 2022. Error correction in asr using sequence-to-sequence models. *arXiv preprint arXiv:2202.01157*.

Yukun Feng, Feng Li, Ziang Song, Boyuan Zheng, and Philipp Koehn. 2022. Learn to remember: Transformer with recurrent memory for document-level machine translation. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1409–1420.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. 2018. Search engine guided neural machine translation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, pages 5133–5140.

Zhiyu Guo and Minh Le Nguyen. 2020. Document-level neural machine translation using bert as context encoder. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 101–107.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

U. Khandelwal, O. Levy, J. Dan, L. Zettlemoyer, and M. Lewis. 2020a. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020b. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings* *of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Bei Li, Hui Liu, Ziyang Wang, Yufan Jiang, Tong Xiao, Jingbo Zhu, Tongran Liu, and Changliang Li. 2020. Does multi-encoder help? a case study on context-aware neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3512–3518.

Shuming Ma, Dongdong Zhang, and Ming Zhou. 2020. A simple and effective unified encoder for document-level machine translation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 3505–3511.

Sameen Maruf, André FT Martins, and Gholamreza Haffari. 2019. Selective attention for context-aware neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3092–3102.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE.

Yunfan Shao, Zhichao Geng, Yitao Liu, Junqi Dai, Hang Yan, Fei Yang, Li Zhe, Hujun Bao, and Xipeng Qiu. 2021. Cpt: A pre-trained unbalanced transformer for both chinese language understanding and generation. *arXiv preprint arXiv:2109.05729*.

Zewei Sun, Mingxuan Wang, Hao Zhou, Chengqi Zhao, Shujian Huang, Jiajun Chen, and Lei Li. 2022. Rethinking document-level neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3537–3548, Dublin, Ireland. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1264–1274, Melbourne, Australia. Association for Computational Linguistics.

Dexin Wang, Kai Fan, Boxing Chen, and Deyi Xiong. 2022a. Efficient cluster-based k-nearest-neighbor machine translation-nearest-neighbor machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2175–2187.

Xiaoqiang Wang, Yanqing Liu, Jinyu Li, Veljko Miljanic, Sheng Zhao, and Hosam Khalil. 2022b. Towards contextual spelling correction for customization of end-to-end speech recognition systems. *arXiv preprint arXiv:2203.00888*.

Xiaoqiang Wang, Yanqing Liu, Sheng Zhao, and Jinyu Li. 2021. A light-weight contextual spelling correction model for customizing transducer-based speech recognition systems. In *Proc. Interspeech 2021*, pages 1982–1986.

Lesly Miculicich Werlen, Dhananjay Ram, Nikolaos Pappas, and James Henderson. 2018. Document-level neural machine translation with hierarchical attention networks. In *EMNLP*.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2021. Memorizing transformers. In *International Conference on Learning Representations*.

Zhengxin Yang, Jinchao Zhang, Fandong Meng, Shuhao Gu, Yang Feng, and Jie Zhou. 2019. Enhancing context modeling with a query-guided capsule network for document-level translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1527–1537.

Zhuoyuan Yao, Di Wu, Xiong Wang, Binbin Zhang, Fan Yu, Chao Yang, Zhendong Peng, Xiaoyu Chen, Lei Xie, and Xin Lei. 2021. Wenet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit. In *Proc. Interspeech*, Brno, Czech Republic. IEEE.

Xunjian Yin, Xinyu Hu, and Xiaojun Wan. 2022. Chinese spelling check with nearest neighbors. *arXiv preprint arXiv:2211.07843*.

Zheng Yuan and Christopher Bryant. 2021. Document-level grammatical error correction. In *Proceedings of the 16th Workshop on Innovative*

Use of NLP for Building Educational Applications, pages 75–84.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, Min Zhang, and Yang Liu. 2018. Improving the transformer translation model with document-level context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 533–542.

Linlin Zhang, Zhirui Zhang, Boxing Chen, Weihua Luo, and Luo Si. 2022. Context-adaptive document-level neural machine translation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6232–6236. IEEE.

Yun Zhao, Xuerui Yang, Jinchao Wang, Yongyu Gao, Chao Yan, and Yuanfu Zhou. 2021. BART Based Semantic Correction for Mandarin Automatic Speech Recognition System. In *Proc. Interspeech 2021*, pages 2017–2021.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. 2021. Adaptive nearest neighbor machine translation. In *ACL/IJCNLP (2)*.

Zaixiang Zheng, Yue Xiang, Shujian Huang, Jiajun Chen, and Alexandra Birch-Mayne. 2020. Toward making the most of context in neural machine translation. In *29th International Joint Conference in Artificial Intelligence*, pages 3983–3989. International Joint Conferences on Artificial Intelligence Organization.

Zexuan Zhong, Tao Lei, and Danqi Chen. 2022. Training language models with memory augmentation. *arXiv preprint arXiv:2205.12674*.

11. Language Resource References

Bu, Hui and Du, Jiayu and Na, Xingyu and Wu, Bengu and Zheng, Hao. 2017. *Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline*. IEEE. Beijing Shell Shell Technology Co.,Ltd., ISLRN 733-251-884-636-1.

Panayotov, Vassil and Chen, Guoguo and Povey, Daniel and Khudanpur, Sanjeev. 2015. *Librispeech: An ASR corpus based on public domain audio books*. IEEE.

Pascale Fung, Shudong Huang, David Graff. 2005. *HKUST Mandarin Telephone Speech, Part 1*. Speecon Project, distributed via ELRA: ELRA-Id S0327, ISLRN 964-004-555-226-5.

A. Appendix

A.1. Datasets

We evaluate our approach on four datasets: AISHELL-1, HKUST, LIBRISPEECH_CLEAN and LIBRISPEECH_OTHER. Table 10 displays statistics and dataset-specific model settings.

A.2. Hyperparameters of the Model

The model configurations and hyperparameters are shown in Table 9. Model checkpoint and model config for pre-training BART are from huggingface[2][3].

seed	2023
optimizer	AdamW
lr	0.00005
weight_decay	0.02
scheduler type	linear
early stop	5
beam search	4
device	A40-48GB GPU
pretrain-model	BART-BASE
kNN layers	6

Table 9: Model configurations and hyperparameters in our experiments.

A.3. Context Case

More cases can be seen in the table 11.

A.4. Evaluation on ChatGPT

Table 8 displays the performance results of ChatGPT, which were obtained using the gpt-3.5-turbo-0301 model. We evaluated ChatGPT in two modes: chatgpt-sent, which inputs a single sentence, and chatgpt-doc, which inputs either an entire document or multiple sentences. The prompts used for each mode are described in detail in Table 12.

[2] https://huggingface.co/facebook/bart-base

[3] https://huggingface.co/fnlp/bart-base-chinese

Datatset	AISHELL-1	HKUST
durations(hours)	178	148.6
sentence number	120,098/14,326/7,716	159,532/3,307/4508
sentence average length	15.41/15.33/15.6	14.30/16.69/14.73
document number	340/20/40	852/22/24
average sentence number	353.23/358.8/358.15	187.24/150.32/187.83
language	zh	zh
ASR	wenet(Yao et al., 2021)	went(Yao et al., 2021)
batch size	128/40/20	64/22/24
datastore size	43120/14840/14920	250160/16560/19920
max seqence length	40	80
Datatset	**LIBRISPEECH_CLEAN**	**LIBRISPEECH_OTHER**
hours	475	507.1
sentence number	132,544/2,621/2,704	32,922/2,940/2865
sentence average length	34.59/20.06/20.12	32.92/17.80/17.78
document number	2,683/87/96	622/90/91
average sentence number	49.42/30.13/27.88	52.93/33.67/31.48
language	en	en
ASR	fairseq(Ott et al., 2019)	fairseq(Ott et al., 2019)
batch size	48/48/48	48/48/48
datastore size	284300/11300/9800	95700/13000/13100
max seqence length	100	100

Table 10: Statistics and dataset-specific model settings of four datasets. Format: train/dev/test.

	Sentence
Context	re enter butler and three footmen who remove the tea things hostess to guest.
Label	in novels the hero has often pushed his meals away untasted but no stage hero would do anything so unnatural as this.
BART	and novels the hero has often pushed his meals away untasted but no steed hero would do anything so unnatural as this.
Ours	in novels the hero has often pushed his meals away untasted but no steed hero would do anything so unnatural as this.
Context	好莱坞当红明星之前曾被盛传将扮演钢铁侠(Iron Man)。
Label	他(He)确实拿下了这个角色。
BART	她(She)确实拿下了这个角色。
Ours	他(He)确实拿下了这个角色。
Context	李斯达(name)表示自己跟周云露并没有深仇大恨。
Label	目前李斯达(name)被关押在朝阳区看守所。
BART	目前李思达(name)被关押在朝阳区看守所。
Ours	目前李斯达(name)被关押在朝阳区看守所。
Context	呼救报警时称有人入屋行凶。
Label	又(and)供称是自己失手杀妻。
BART	有(have)传称是自己失手杀妻。
Ours	又(and)供称是自己失手杀妻。

Table 11: Cases.

Chatgpt-sent	
Prompt	你是一个通用领域的语音识别后纠错系统，主要纠正语音识别过程中由于发音相似和遗漏识别导致的错误，且纠错前后句子长度接近。首先逐个地检测每个字是否错误，并逐个地纠正对应的错误字符，输入中大部分字是正确的；然后系统输出完整的纠错后的句子。系统输出格式为：“纠错结果：”。系统要尽可能保持输入句子的结构，尽可能少修改，不能过度纠正，系统只输出最后的纠错结果。给下面句子纠错:
Translation	You are a general-purpose domain post-speech recognition error correction system, which mainly corrects errors caused by similar pronunciation and missed recognition during speech recognition, and the sentence length before and after error correction is close. First detects each word one by one for errors and corrects the corresponding incorrect characters one by one, with most of the words in the input being correct;Then the system outputs the complete corrected sentence. The system output format is: "Error correction result:" . The system should keep the structure of the input sentence as much as possible, modify as little as possible, and not over-correct, and the system only outputs the final error correction result. Correct the following sentence:
Chatgpt-doc	
Prompt	你是一个通用领域的文档级语音识别后纠错系统，主要纠正语音识别过程中由于发音相似和遗漏识别导致的错误，且纠错前后句子长度接近。另外你要考虑文档中的上下文信息来纠正错误。首先逐个地检测每个字是否错误，并逐个地纠正对应的错误字符，输入中大部分字是正确的；然后系统输出完整的纠错后的句子。系统输出格式为：“纠错结果：”。系统要尽可能保持输入句子的结构，尽可能少修改，不能过度纠正，系统只输出最后的纠错结果。给下面句子纠错:
Translation	You are a general-purpose domain **document-level post-speech recognition error correction system** that corrects errors caused by similar pronunciation and missed recognition during speech recognition, and the sentence lengths before and after correction are close. In addition you want to consider the contextual information in the document to correct the errors. First detecting each word one by one whether it is wrong and correcting the corresponding wrong character one by one, most of the words in the input are correct;Then the system outputs the complete corrected sentence. The system output format is: "Error correction result:". The system should keep the structure of the input sentence as much as possible, modify as little as possible, and not over-correct, and the system only outputs the final error correction result. Correct the following sentences:

Table 12: Sentence-level and document-level chatgpt prompt.