Lossy Text Compression

December 22, 2024

Simone Giovagnoni 1932180
https://github.com/StruzzoStorpio/DLAI_project

Abstract

This report presents the development and implementation of a lossy text compression technique aimed at reducing the size of text files. Lossy text compression differs from lossless methods in that the reconstructed data is an approximation of the original, allowing for a higher compression ratio.The core idea involves applying a series of preprocessing steps, such as spelling reduction, followed by embedding quantization to optimize storage efficiency.

1. Introduction

In recent years, the number of text data available online is growth enormously. Generating the problem of having enough storage space to save this amount of data. Data compression is playing a crucial role in this situation, saving storage space efficiently and increasing the transmission speed. Text compression could be divided into two categories, lossless and lossy.

1.1. Lossless text compression

Lossless text compression allows reconstructing the data without losing information. This is obviously a strong positive side of this method. Exist three main types of lossless text compression: Statistical methods, Dictionary methods, Substitution methods.

1.2. Lossy text compression

Lossy text compression generally allows higher compression ratio, but the reconstruction of the data is an approximation of the original data. This means that the reconstruction is not exactly equal to the source text.

Email: Simone Giovagnoni <giovagnoni.1932180@studenti.uniroma1.it>.

Deep Learning and Applied AI 2024, Sapienza University of Rome, 2nd semester a.y. 2023/2024.

2. Related work

In this section are shown all the related works, that were studied and used during the development of this project.

2.1. Vec2Text(Morris et al., 2023)

This paper proposes a methodology to generate numerical embeddings from text, and transform the embeddings back into text. Text embedding models learn to map text sequences to embedding vectors. Embedding vectors are useful because they encode some notion of semantic similarity: inputs that are similar in meaning should have embeddings that are close in vector space.
But the greatest challenge in this research field is the correct reconstruction of the text from the embeddings. The authors present a method called Vec2Text, which uses a multi-step iterative approach to accurately recover text from its embeddings, achieving near-perfect reconstruction in many cases. The method successfully recovers 92% of 32-token text inputs exactly, demonstrating the significant amount of information that embeddings can reveal about the original text.

2.2. Text embedding quantization(Shakir et al., 2024)

The article discusses embedding quantization techniques, specifically binary and scalar quantization, to enhance the speed, reduce the memory usage, and lower the costs of retrieval tasks. By converting embeddings from float32 to binary or int8, it is possible to significantly reduce resource requirements while maintaining most of the original performance. The article highlights experiments showing up to 45x speedups and notable savings in storage without substantial losses in retrieval accuracy.

3. Method

It is possible to distinguish different sections of this project. The initial text reduction, the transformation into numerical embeddings, the quantization of the embeddings, the dequantization, the reconstruction of the texts and the evaluation phase. The core idea of this project was to initially do a spelling reduction on the input text, then using vec2text

quantize the embeddings from float32 to int8, this means that, normally the words are represented using 32-bit float number, applying this quantization method, we try to map those texts into int8, so representing the words using 8-bit integer number, which requires less space.

3.1. Text reduction

Initially, the input text receives an initial reduction that could be seen as a pre-processing, in order to already reduce the number of bytes used. This approach has been taken after a careful study of the compression methods proposed in the article "Semantic and Generative models for lossy text compression"(Witten et al., 1994). Firstly, all the punctuations are removed, this, on average, can reduce the dimension of the text of +/- 3%. Then another reduction approach is used, thinking about the English slang, transform some parts of words into numbers("to" $\rightarrow$ 2 ; "tree" $\rightarrow$ 3 ; "for" $\rightarrow$ 4 ; "ate" $\rightarrow$ 8). In this way, the words are still readable and understandable and the reduction is of another +/- 2%.
Then I tried to remove the double letters, using just one letter. But this reduced enormously the quality of the reconstruction.

3.2. From text to embeddings

To quantize the text input, it is necessary to transform the text into numerical embeddings, this means that each word needs to be transformed into float32 values.
To do so, has been used the vec2text(Morris et al., 2023) pre-trained model, which can embed text into float32 and then invert the embeddings back to text.

3.3. Embeddings quantization

After different researches on quantization methods for lossy text compression, the best solution was to map the float32 embeddings into int8 embeddings, this allows to reduce the dimensionality of the embeddings to almost the 75%, so if we quantize embeddings of 1000 bytes with the quantization to int8 becomes 250 bytes, with an information loss close to 0%.

3.4. Embeddings dequantization and reconstruction

Then to evaluate the quality of the quantization, it is necessary to reconstruct the text from the embeddings. Foremost, dequantize from int8 back to float32. Then apply the existing method of vec2text(Morris et al., 2023) to invert the embeddings into text.

3.5. Evaluation

To evaluate the quality of the quantization, the original text is compared with the reconstructed version. The evaluation metrics used are: Cosine similarity, Euclidean Distance, BLEU score, ROUGE score, Edit Distance, Jaccard Distance.

4. Results

The system has been tested on a list of texts with different types of sentences, questions, short sentences, literary excerpt, scientific and technical sentences, complex texts and more.
This list of sentences initially has a dimensionality of 1223 bytes, after the removal of the punctuation decreases to 1193 ($\simeq 97\%$). Then, thanks to the contractions of transforming part of words into numbers, the number of bytes decreases to 1174 ($\simeq 95\%$). Then after the embeddings' transformation, the quantization to int8 brings the reduction to 25% of 95%, so the number of bytes used for this compressed text is $\simeq 23,5\%$ of the starting text.
The results of the evaluation are shown in the table below. The first column of results shows the result obtained applying only punctuation removal before quantization, the second with also the contraction into numbers, the third one including also the double consonants' removal.

Table 1. Reconstruction similarity

Metrics	Results 1	Results 2	Results 3
Cosine Similarity	0.9991	0.9990	0.9988
Euclidean Distance	0.0632	0.0632	0.0635
BLEU -avg-	0.5165	0.4194	0.4022
ROUGE-1	0.8906	0.8470	0.7816
ROUGE-2	0.7500	0.6497	0.5827
ROUGE-L	0.8323	0.7860	0.7351
Edit Distance -avg-	32,69	33	35
Jaccard distance -avg-	0.7159	0.6537	0.5828

Between all those metrics, the most reliable, that can clearly shows if the reconstruction is close to the original are the BLEU, the Jaccard distance and the three ROUGE scores. The semantic similarity scores, the euclidean distances and the edit distances, as you can see, are very similar even if the three reconstructed texts have substantial differences from a human point of view. From the results is clear that the removal of double consonants causes significant damages to the reconstruction process

5. Discussion and conclusions

The results of this study confirm the potential of lossy text compression techniques to significantly reduce the storage and transmission demands of large text datasets. By utilizing a multi-stage process that includes preprocessing, text-to-embedding conversion, and quantization, the system was able to compress text by over 75% while maintaining high reconstruction quality.

Notably, the approach that involved transforming parts of words into numbers proved effective, contributing to further size reductions without sacrificing readability. However, the removal of double consonants led to noticeable quality degradation.

References

Morris, J. X., Kuleshov, V., Shmatikov, V., and Rush, A. M. Text embeddings reveal (almost) as much as text, 2023. URL https://arxiv.org/abs/2310.06816.

Shakir, A., Aarsen, T., and Lee, S. Binary and scalar embedding quantization for significantly faster cheaper retrieval. *Hugging Face Blog*, 2024. https://huggingface.co/blog/embedding-quantization.

Witten, I. H., Bell, T. C., Moffat, A., Nevill-Manning, C. G., Smith, T. C., and Thimbleby, H. Semantic and Generative Models for Lossy Text Compression. *The Computer Journal*, 37(2):83–87, 01 1994. ISSN 0010-4620. doi: 10.1093/comjnl/37.2.83. URL https://doi.org/10.1093/comjnl/37.2.83.