Technical University of Cluj-Napoca
Big Data Seminar

Image Captioning

Presented by Sara-Ioana Voicu

Overview

01 Introduction

02 Dataset

03 Bibliographic Research

04 Proposed Design

05 Implementation

06 Results

07 Improvements and Conclusion

Introduction

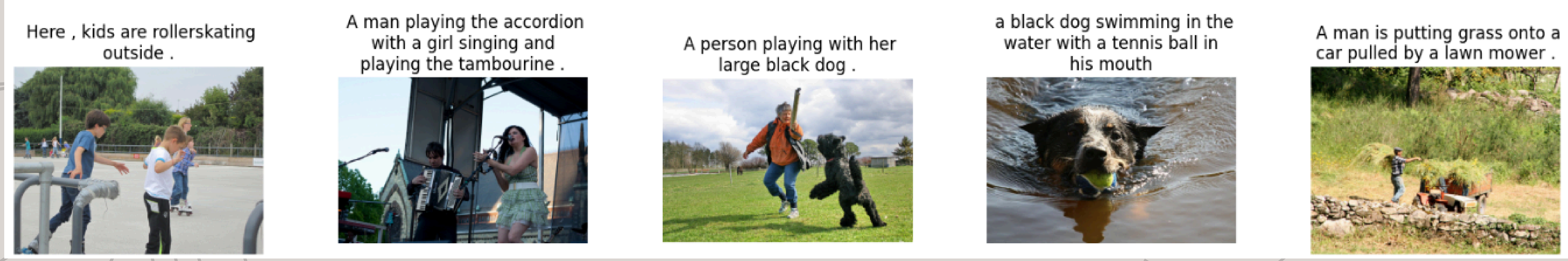
Image captioning represents the task of generating sentences to describe the visual content of an image.
Unlike human beings who can easily describe pictures, computers must combine expertise from multiple fields:

- Image processing
- Computer vision
- Natural language processing

Dataset

Flickr Dataset - collection of images and captions

- Flickr8k: 8,000 images
- Flickr30k: 30,000 images
- each image has 5 different descriptions
- similar datasets for image captioning: MS COCO (300, 000 images), Google Open Images (around 9 million images)



Here , kids are rollerskating outside .

A man playing the accordion with a girl singing and playing the tambourine .

A person playing with her large black dog .

a black dog swimming in the water with a tennis ball in his mouth

A man is putting grass onto a car pulled by a lawn mower .

Bibliographic Research

The article [1] explores approaches to generating textual descriptions of images using deep learning models. It focuses on encoder-decoder architectures and proposes three approaches for image captioning:

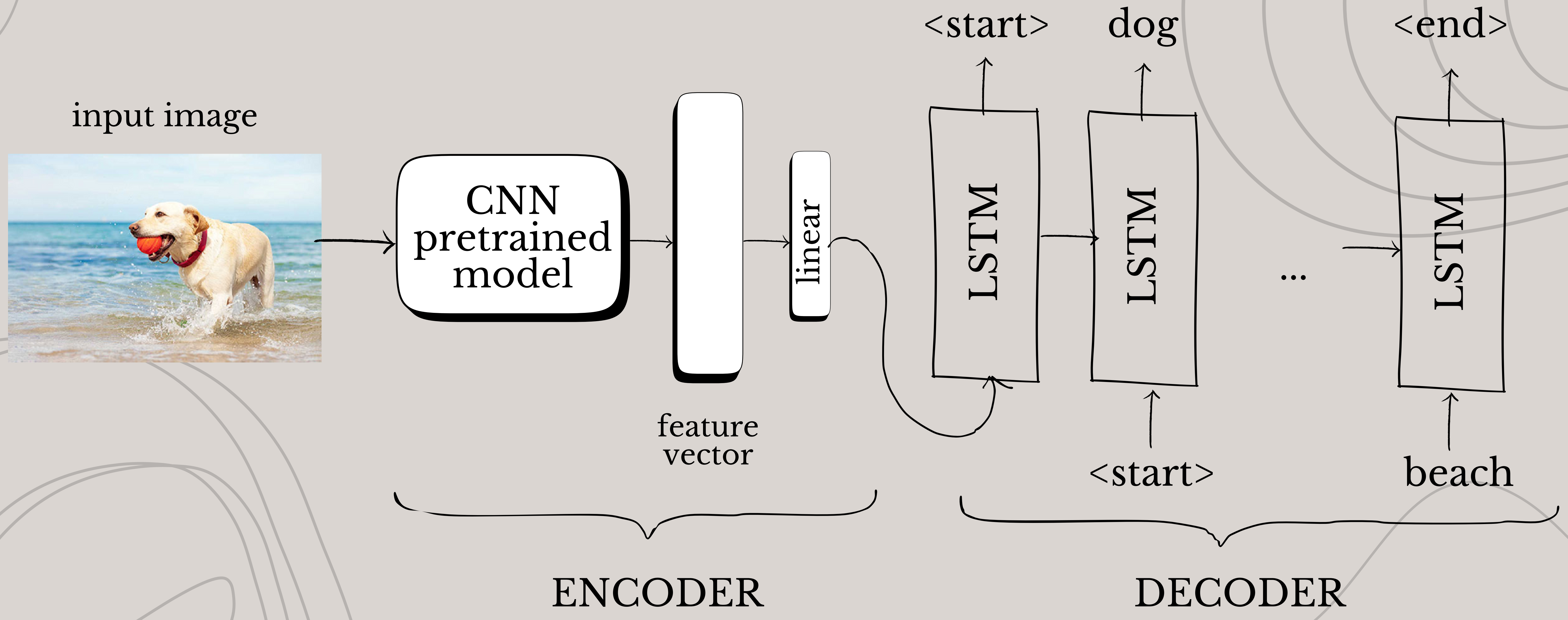
- CNN - RNN (LSTM)
- CNN - CNN
- Reinforcement based learning

Challenges discovered in the article:

- create semantically and syntactically accurate captions
- need of advanced models
- understand object relationships, context, and actions in images

Metrics were calculated using the following methods: BLUE, METEOR, ROUGE, CIDEr, and SPICE

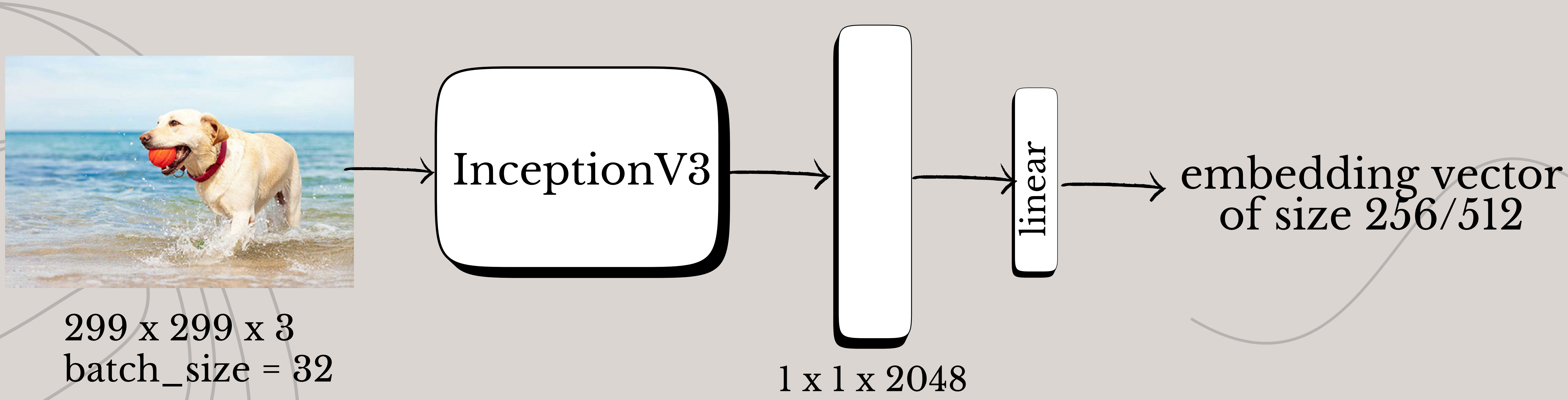
Proposed Design



Implementation

Encoder:

- CNN architecture
- InceptionV3 pre-trained model: inception modules that contain Convolutional + Pooling layers
- fine-tuning
- Linear layer



InceptionV3

linear

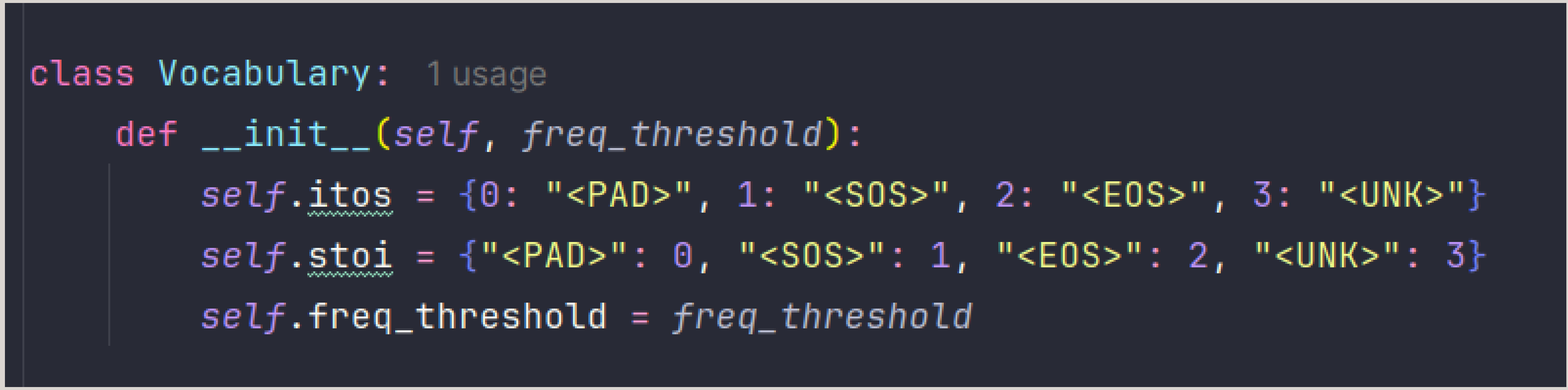
embedding vector of size 256/512

299 x 299 x 3
batch_size = 32

1 x 1 x 2048

Vocabulary class:

- converts each word into a numeric value



```python
class Vocabulary:
    def __init__(self, freq_threshold):
        self.itos = {0: "<PAD>", 1: "<SOS>", 2: "<EOS>", 3: "<UNK>"}
        self.stoi = {"<PAD>": 0, "<SOS>": 1, "<EOS>": 2, "<UNK>": 3}
        self.freq_threshold = freq_threshold
```

First dictionary:

- itos = index to string
- maps an index to its corresponding word
- used for decoding the final predictions

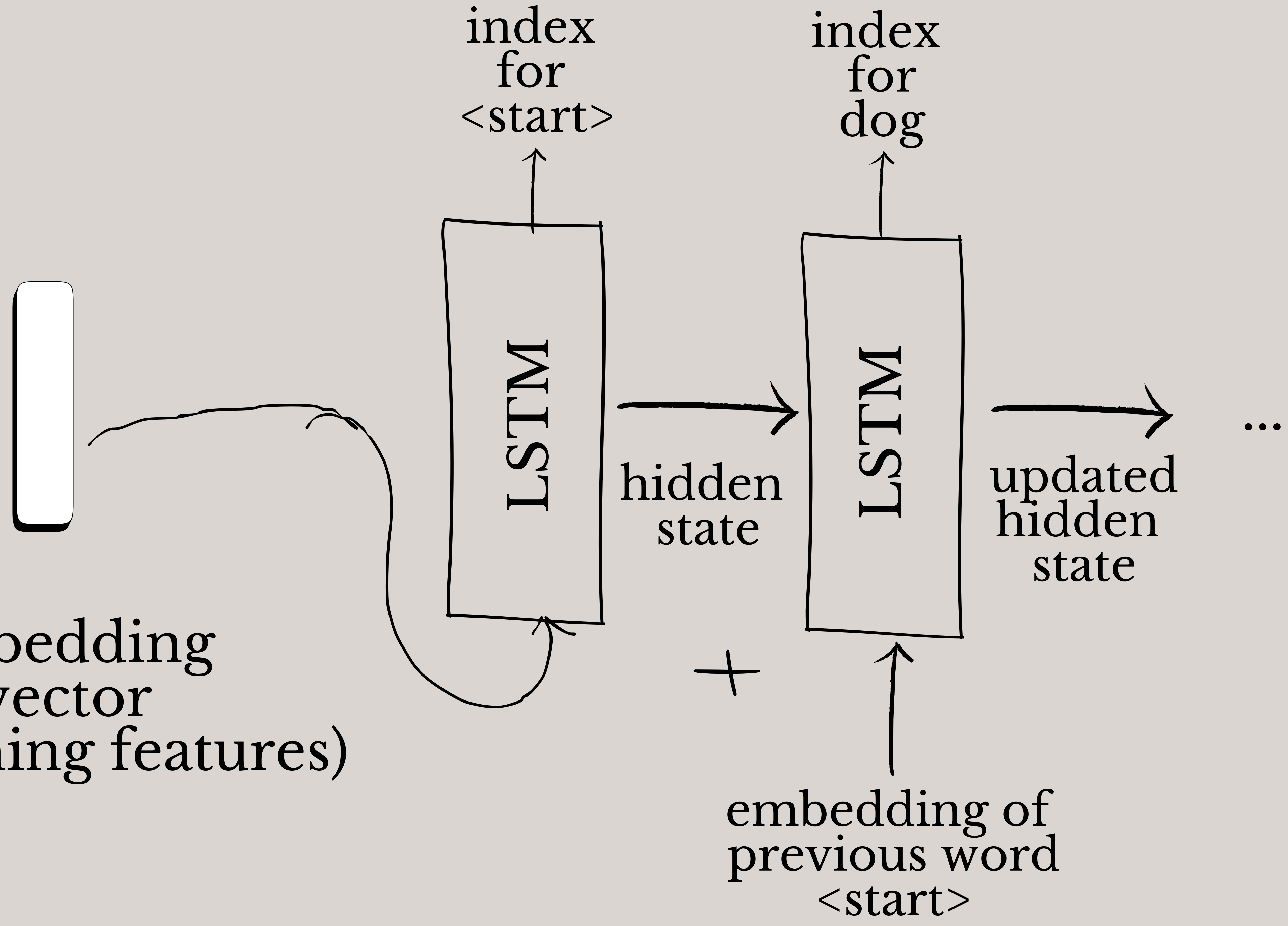
Second dictionary:

- stoi = string to index
- maps a word to its corresponding index
- used for LSTM model (words are represented as integers during processing)

Threashold = the minimum number of occurrences a word must have to be included in the vocabulary

Decoder:

- RNN architecture [2]
- LSTM model [3]



index for <start>

index for dog

LSTM

LSTM

hidden state

updated hidden state

...

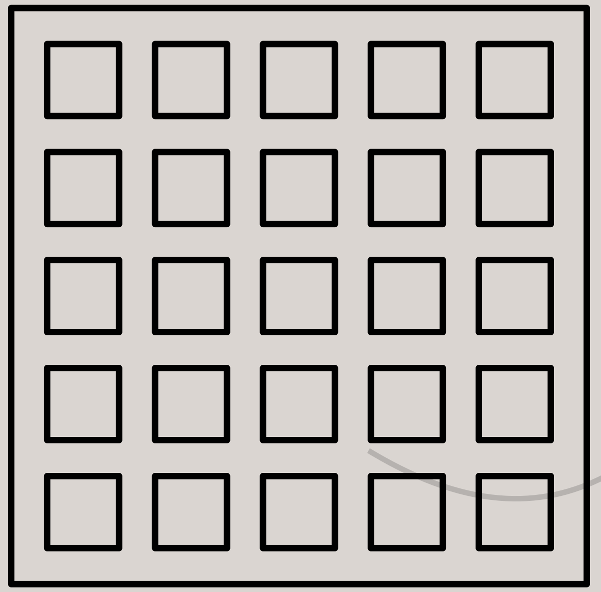
embedding vector (containing features)

+

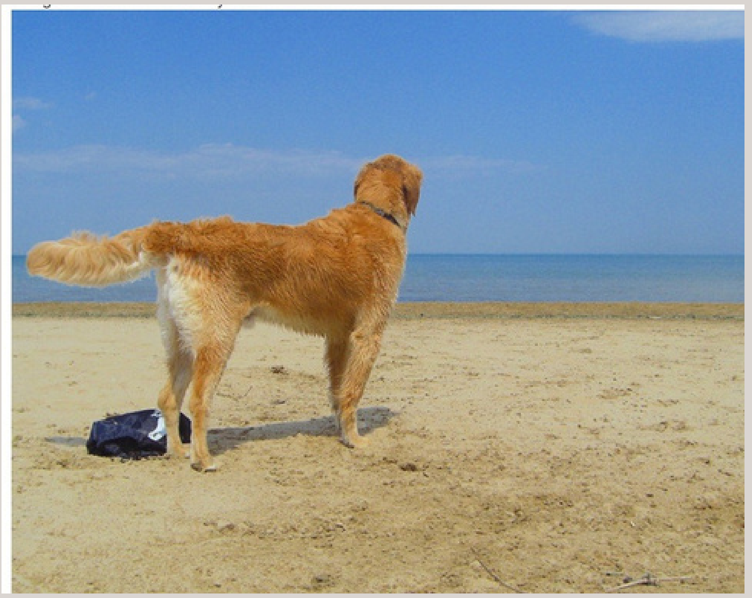
embedding of previous word <start>

How do we get the embedding of a word?

- matrix
- rows: vocabulary_size
- columns: embed_size
- each row represents the embedding of a word



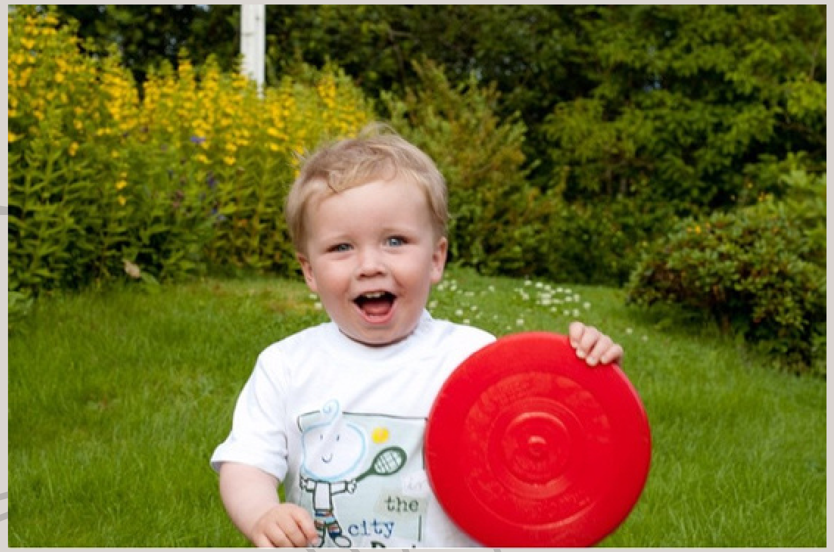
Results



Flickr8k: <SOS> a brown dog is running through the water. <EOS>

Flickr8k: <SOS> a dog is running in the ocean with a stick in its mouth . <EOS>
(increased embed_size)

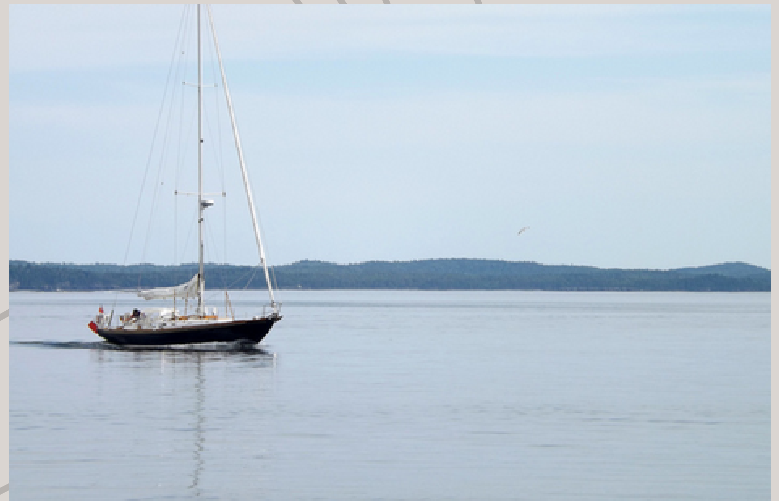
Flickr30k: <SOS> a dog is running through the water . <EOS>



Flickr8k: <SOS> a little boy in a blue shirt is playing with a soccer ball . <EOS>

Flickr8k: <SOS> a little girl in a pink shirt is running through a flowered field . <EOS>
(increased embed_size)

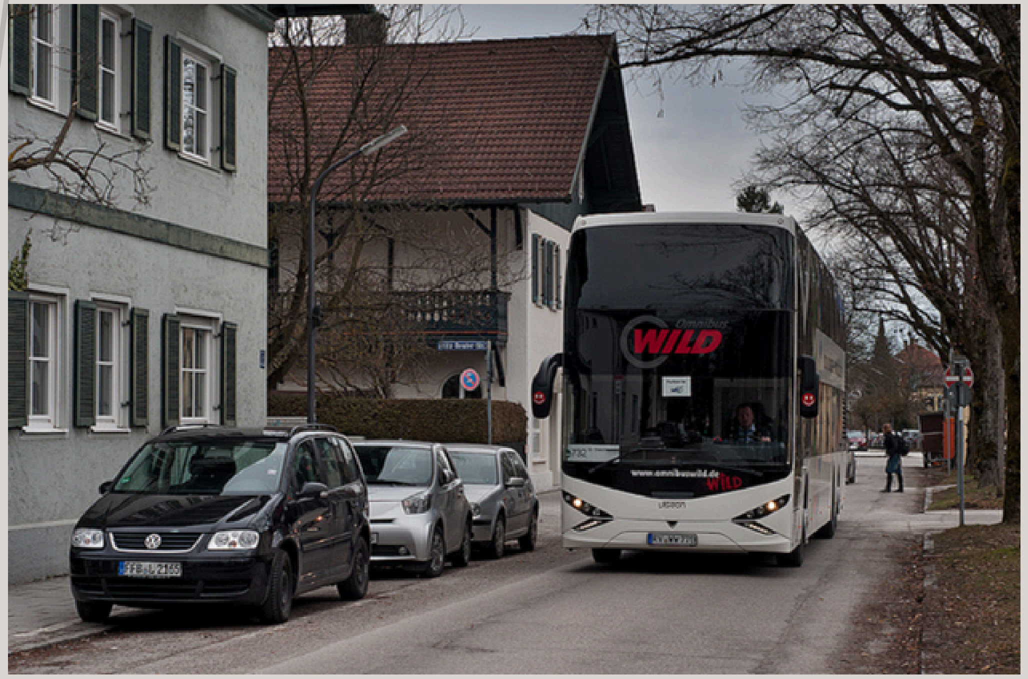
Flickr30k: <SOS> a young girl in a pink shirt is playing with a toy . <EOS>



Flickr8k: <SOS> a man is rowing a canoe through the water . <EOS>

Flickr8k: <SOS> a man in a blue shirt is rowing a boat <EOS>
(increased embed_size)

Flickr30k: <SOS> a man is standing on a boat in the middle of a lake . <EOS>

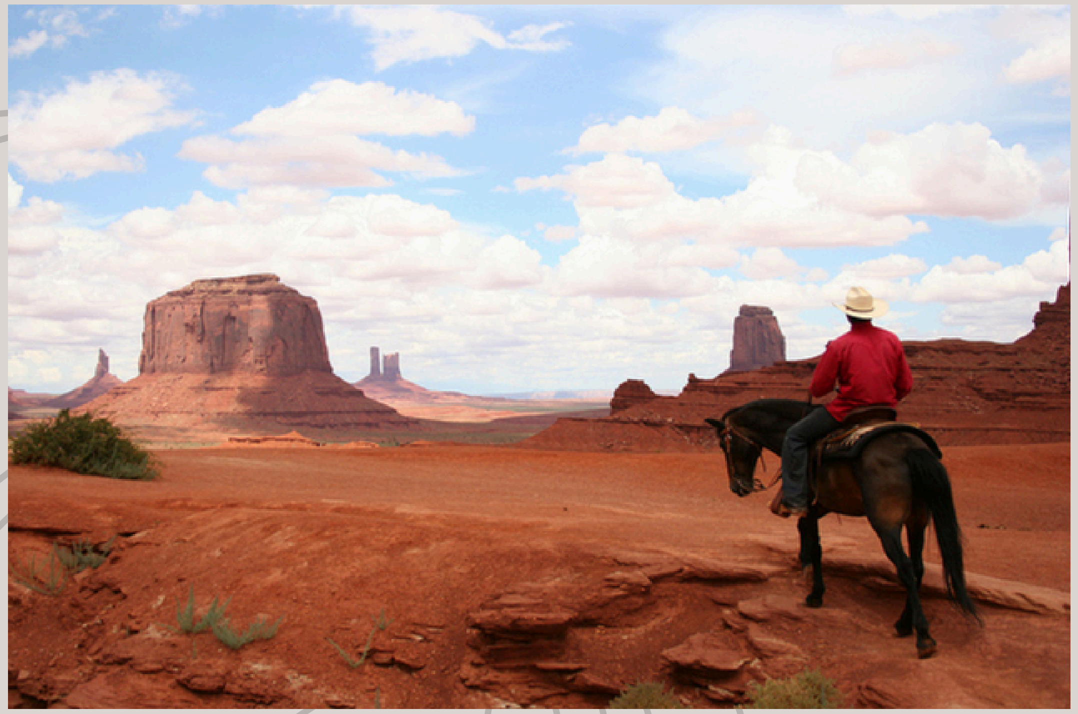


Flickr8k: <SOS> a man in a black shirt and jeans is standing in front of a <UNK> . <EOS>

Flickr8k: <SOS> a man walks in the city . <EOS>

(increased embed_size)

Flickr30k: <SOS> a man in a black shirt is standing in front of a red bus . <EOS>

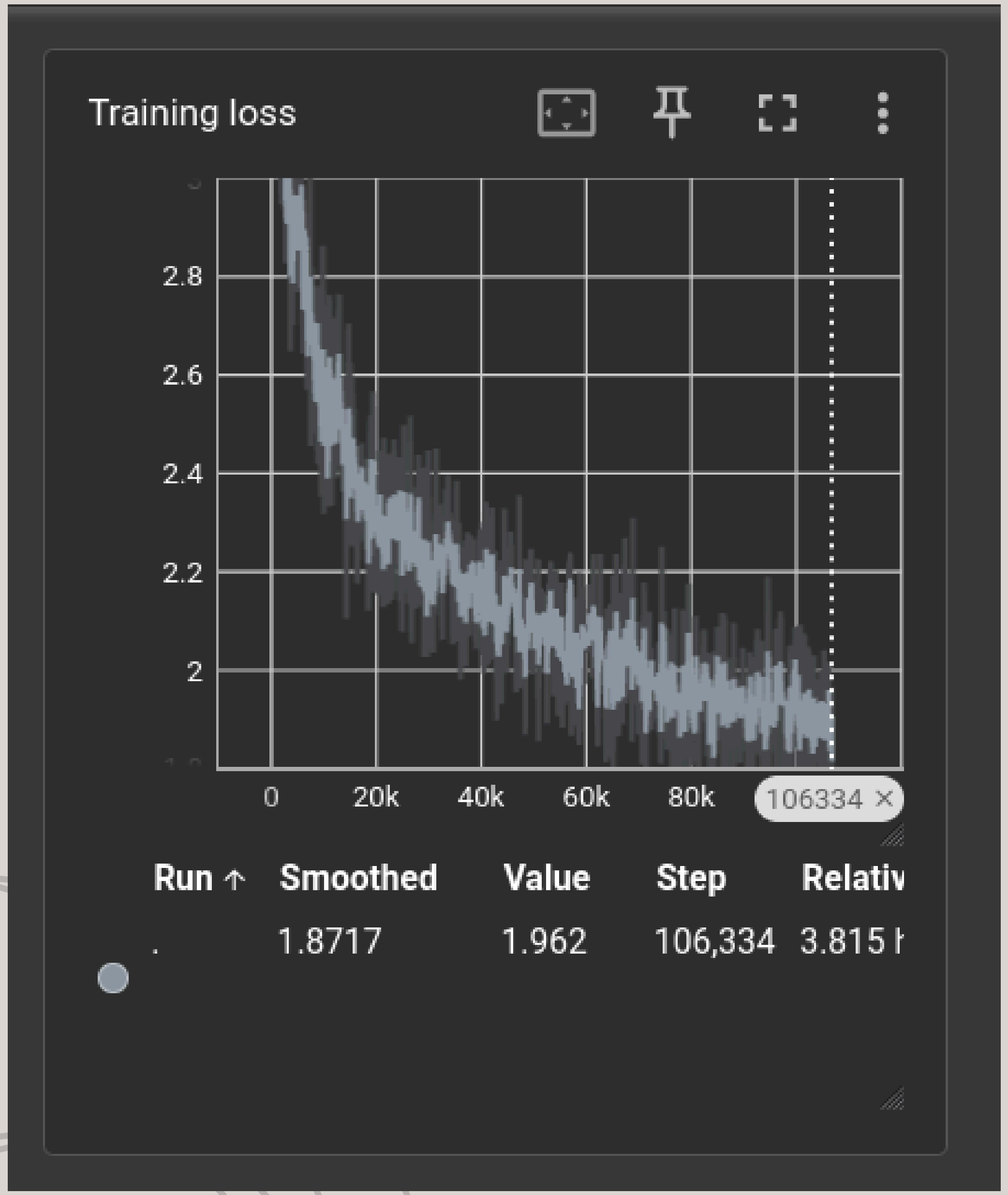


Flickr8k: <SOS> a man and a dog are standing on a beach . <EOS>

Flickr8k: <SOS> a man in a cowboy hat is riding a brown horse over a rocky shore . <EOS>

(increased embed_size)

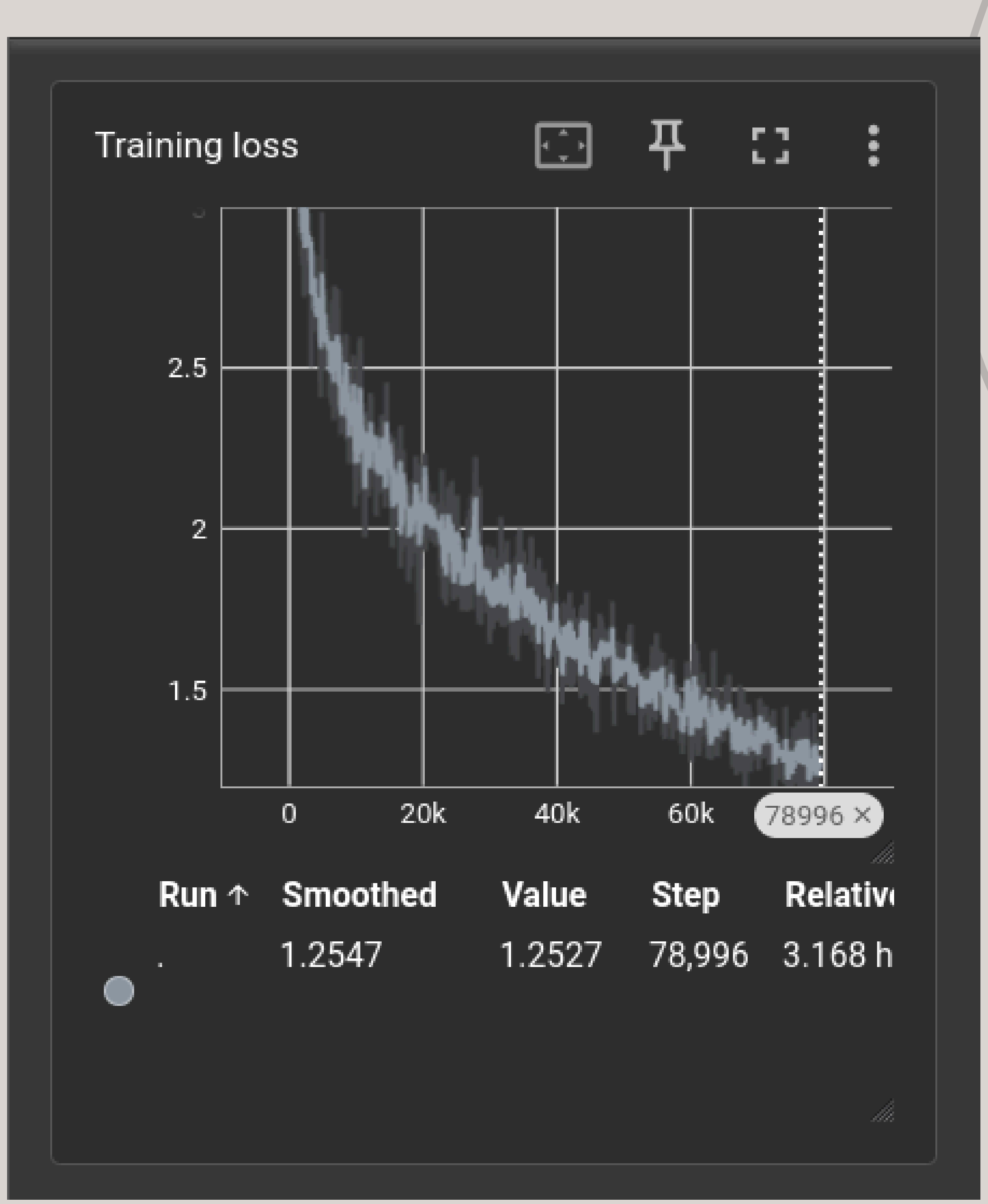
Flickr30k: OUTPUT: <SOS> a man in a shirt is riding a horse on a dirt road . <EOS>



Run ↑	Smoothed	Value	Step	Relativ
.	1.8717	1.962	106,334	3.815 h

Flickr8k

loss = 1.96
4 hours of training
100 epochs



Run ↑	Smoothed	Value	Step	Relativ
.	1.2547	1.2527	78,996	3.168 h

Flickr8k
(improved embe_size)

loss = 1.25
3 hours of training
78 epochs

Flickr30k

more than 10 hours
46 epochs

Improvements and Conclusion

Improvements:

- use the Flickr30k dataset for better diversity and larger sample size.
- increase the embedding size, the number of LSTM layers, and the number of training epochs for a more robust model
- change the pre-trained CNN model
- explore a CNN-CNN design
- Implement a transformer-based architecture
- OpenAI CLIP technique

Conclusion: image captioning is a complex task that combines many domains. With the help of a big dataset and powerful advanced models, good results and significant improvements can be obtained.

Bibliography

- [1] S. Liu, L. Bai, Y. Hu, and H. Wang, "Image Captioning Based on Deep Neural Networks," College of Systems Engineering, National University of Defense Technology, 2018.

- [2] Recurrent Neural Networks (RNNs). Available on YouTube: https://www.youtube.com/watch?v=AsNTP8Kwu80

- [3] Long Short-Term Memory (LSTM). Available on YouTube: https://www.youtube.com/watch?v=YCzL96nL7j0

<SOS> Thank you for your attention! <EOS>