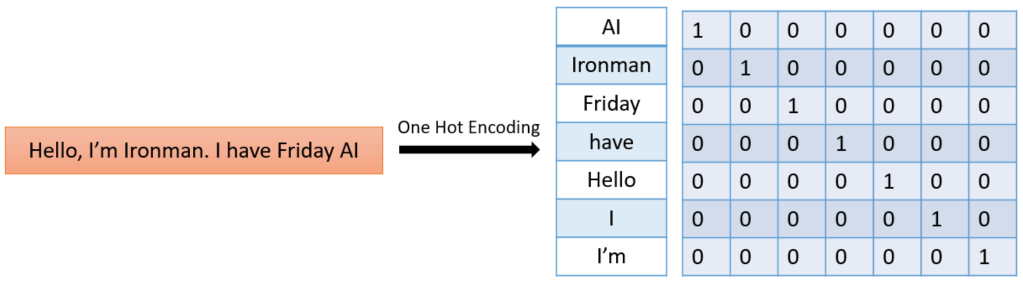
Report on Card 6 - Prática: Embedding (II)

Marcus Vinicius Oliveira Nunes

1. Activity Description

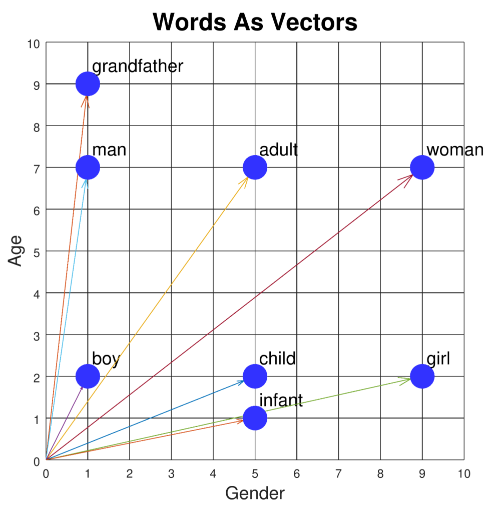
1.1 Introduction to Embeddings

Before the word embedding technique was around, there were other techniques that were used to represent words/texts in a way that machines can understand, one hot encoding is one of the methods used to perform this. One hot encoding represented each word in a sparse vector that had the size of the sentence, the word was represented by the number "1". The problem with this technique was that it couldn't capture the relations between words and couldn't tell if words like "boy" and "man" were similar.



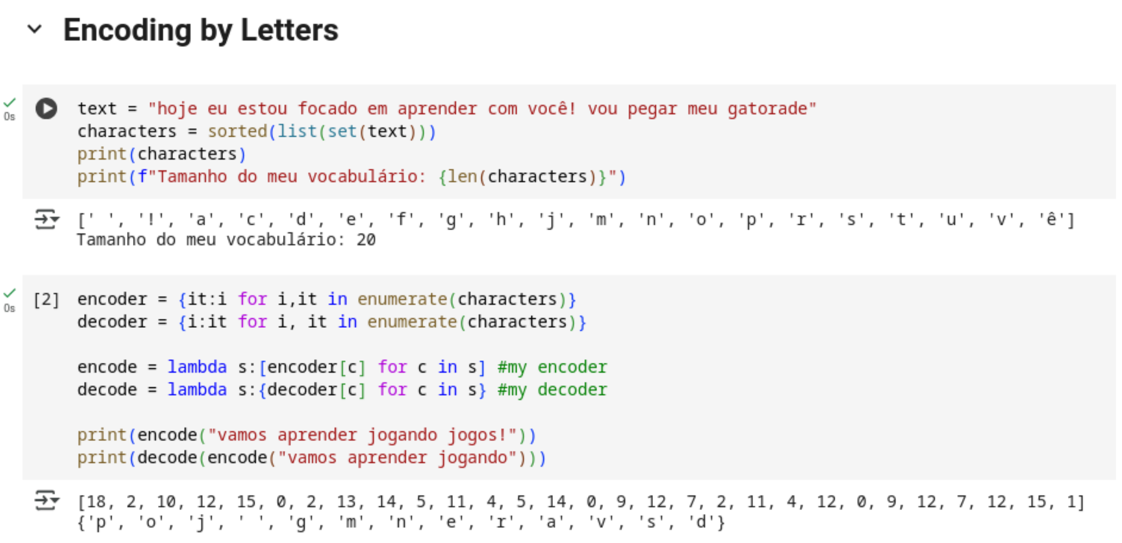
AI	1	0	0	0	0	0	0
Ironman	0	1	0	0	0	0	0
Friday	0	0	1	0	0	0	0
have	0	0	0	1	0	0	0
Hello	0	0	0	0	1	0	0
I	0	0	0	0	0	1	0
I'm	0	0	0	0	0	0	1

Word embedding is a way of representing text into a multidimensional vector that captures semantic relationships between words. In the example below we can see that the words that have a higher level of similarity are closer to each other and the ones that are more different are far away from each other. This approach is useful because it helps the model understand the relations between words, which can help in tasks such as level of similarity between texts.



1.2 Creating a model from scratch (00:00 - 15:40)

This slice of the video is about the process of encoding, decoding and the way the models predict the words. In order to make a computer understand and process words, it's necessary to encode the text since the computer can't understand human language, only numbers. Encoding works by giving a number to associate with a certain character/ word making it into a token. With texts we can create an array of words. Here is an encoder that works by letters:



Encoding by Letters

```
text = "hoje eu estou focado em aprender com você! vou pegar meu gatorade"
characters = sorted(list(set(text)))
print(characters)
print(f"Tamanho do meu vocabulário: {len(characters)}")
```

```
[' ', '!', 'a', 'c', 'd', 'e', 'f', 'g', 'h', 'j', 'm', 'n', 'o', 'p', 'r', 's', 't', 'u', 'v', 'ê']
Tamanho do meu vocabulário: 20
```

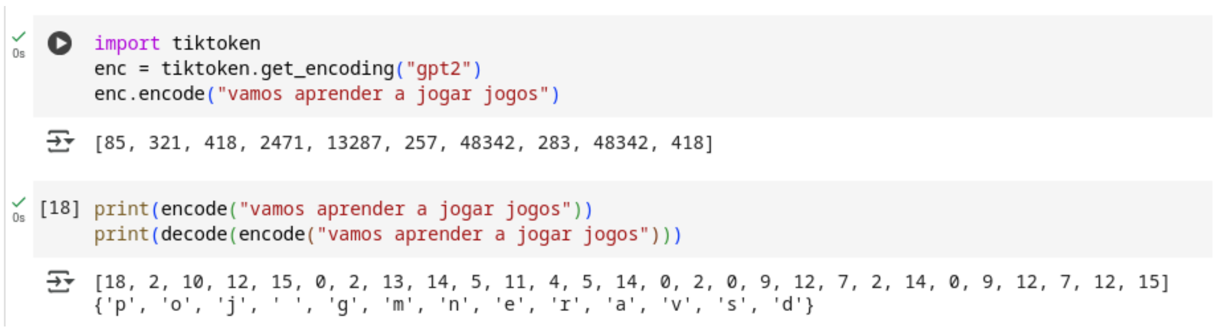
```
[2] encoder = {it:i for i,it in enumerate(characters)}
decoder = {i:it for i, it in enumerate(characters)}

encode = lambda s:[encoder[c] for c in s] #my encoder
decode = lambda s:{decoder[c] for c in s} #my decoder

print(encode("vamos aprender jogando jogos!"))
print(decode(encode("vamos aprender jogando")))
```

```
[18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 9, 12, 7, 2, 11, 4, 12, 0, 9, 12, 7, 12, 15, 1]
{'p', 'o', 'j', ' ', 'g', 'm', 'n', 'e', 'r', 'a', 'v', 's', 'd'}
```

Since encoding by letters requires more processing power, it's better to do it by words. Below we can see a more efficient tokenizer that tokenizes by words, the array's size is smaller, and processing will be more efficient.



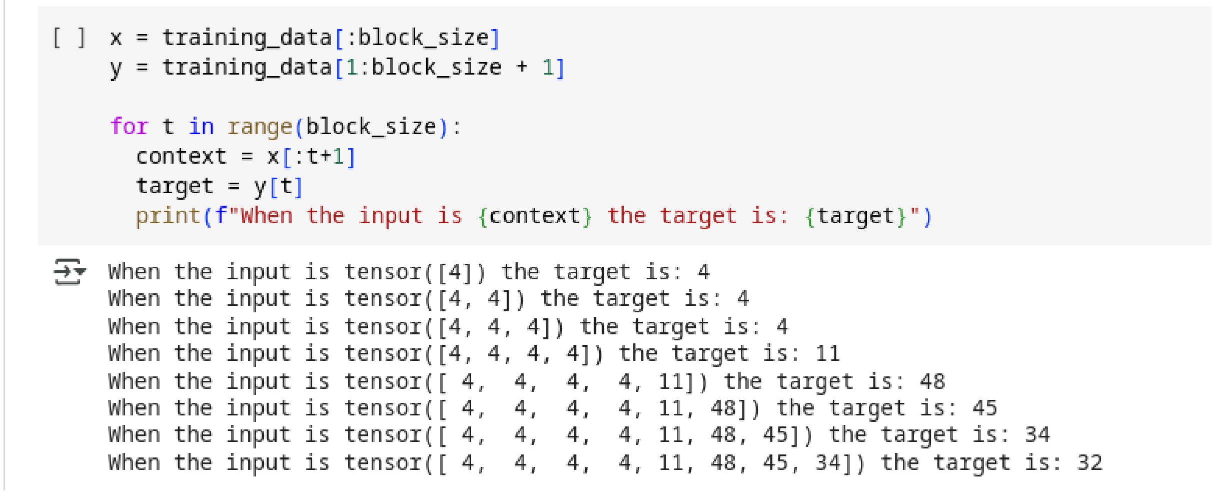
```
import tiktoken
enc = tiktoken.get_encoding("gpt2")
enc.encode("vamos aprender a jogar jogos")
```

```
[85, 321, 418, 2471, 13287, 257, 48342, 283, 48342, 418]
```

```
[18] print(encode("vamos aprender a jogar jogos"))
print(decode(encode("vamos aprender a jogar jogos")))
```

```
[18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 2, 0, 9, 12, 7, 2, 14, 0, 9, 12, 7, 12, 15]
{'p', 'o', 'j', ' ', 'g', 'm', 'n', 'e', 'r', 'a', 'v', 's', 'd'}
```

Following this, the video talks about the way that the models predict. GPT learns from predicting characters that come after another, in a left to right sequence; this approach can be seen both in the video given by the card and in the main LAMIA's machine learning bootcamp. Here is an example from the building chat gpt from scratch video:

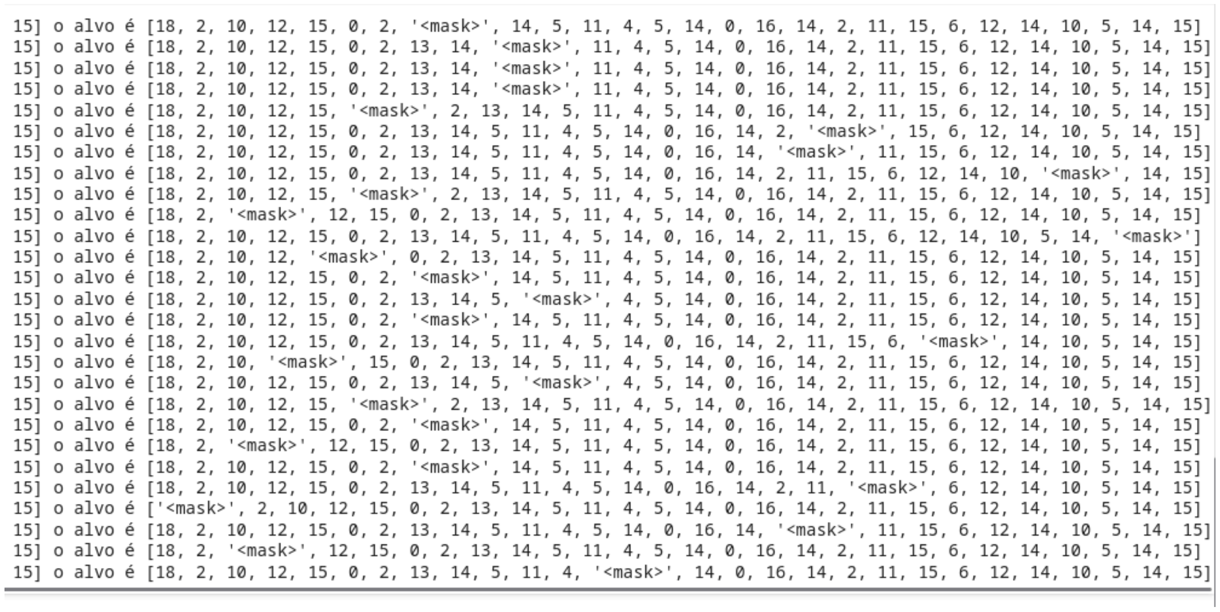


```python
x = training_data[:block_size]
y = training_data[1:block_size + 1]

for t in range(block_size):
  context = x[:t+1]
  target = y[t]
  print(f"When the input is {context} the target is: {target}")
```

```
When the input is tensor([4]) the target is: 4
When the input is tensor([4, 4]) the target is: 4
When the input is tensor([4, 4, 4]) the target is: 4
When the input is tensor([4, 4, 4, 4]) the target is: 11
When the input is tensor([ 4,  4,  4,  4, 11]) the target is: 48
When the input is tensor([ 4,  4,  4,  4, 11, 48]) the target is: 45
When the input is tensor([ 4,  4,  4,  4, 11, 48, 45]) the target is: 34
When the input is tensor([ 4,  4,  4,  4, 11, 48, 45, 34]) the target is: 32
```

It starts from the first token to the last. In this example the sentences are stored in blocks with 8 slots for each token. BERT(Bidirectional Encoder Representations from Transformers), on the other hand uses a different approach, instead of using a left to right approach it uses a random approach, by trying to predict tokens from any position.



```
15] o alvo é [18, 2, 10, 12, 15, 0, 2, '<mask>', 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, '<mask>', 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, '<mask>', 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, '<mask>', 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, '<mask>', 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, '<mask>', 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, '<mask>', 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, '<mask>', 14, 15]
15] o alvo é [18, 2, 10, 12, 15, '<mask>', 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, '<mask>', 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, '<mask>']
15] o alvo é [18, 2, 10, 12, '<mask>', 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, '<mask>', 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, '<mask>', 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, '<mask>', 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, '<mask>', 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, '<mask>', 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, '<mask>', 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, '<mask>', 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, '<mask>', 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, '<mask>', 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, '<mask>', 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, '<mask>', 6, 12, 14, 10, 5, 14, 15]
15] o alvo é ['<mask>', 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, '<mask>', 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, '<mask>', 12, 15, 0, 2, 13, 14, 5, 11, 4, 5, 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
15] o alvo é [18, 2, 10, 12, 15, 0, 2, 13, 14, 5, 11, 4, '<mask>', 14, 0, 16, 14, 2, 11, 15, 6, 12, 14, 10, 5, 14, 15]
```

2. Conclusion

In this card it was pointed out what is word embedding and why it is important to the NLP field, with this approach it's possible to make the model understand the context of the sentences and the relations between words through vectors in a multidimensional vector. It was also pointed out how GPT and BERT word, how they predict.

3. References

https://medium.com/@davidlfliang/intro-getting-started-with-text-embeddings-using-bert-9f8c3b98dee6

O que são Word Embeddings? | Processamento de Linguagem Natural | Leona…

Let's build GPT: from scratch, in code, spelled out.