Prática: Modelos Generativos (III)

Eduardo Prasniewski

1 AutoEncoders e Generative Adversarial Networks

1.1 AutoEncoders

São designados a fim de realizar uma compressão e reconstrução de determinando dado, sendo assim suas características principais são armazenadas em um vetor latente, e posteriormente somente com esse vetor é possível voltar a imagem original. É com posto pelo encoder, que realiza a compressão do dado até o vetor latente e o decoder, que faz o processo inverso. Para a camada latente é realizado a desparametrização, fazendo com que idealmente dois dados sejam treinados e um valor episolon fixo, formando assim uma distribuição normal de probabilidade.

Link Colab: `https://colab.research.google.com/drive/1xYErzgAFpLlk97ua1sMCosCu2BkZpi75?usp=sharing`

1.2 Generative Adversarial Networks

Já as GAN's são compostas basicamente de duas redes neurais, a geradora e a discriminadora. O objetivo é que a geradora gere um dado que a discriminadora não consegue distinguir assertivamente se o que foi gerado é real ou falso. Sendo assim, a função de perda da geradora tende a minizar e a da discriminadora a aumentar. É amplamente utilizada para realizar a criação de imagens, músicas, texto sintéticos etc. Porém tendem a ser muito instáveis no seu processo de treinamento.

Link Colab: `https://colab.research.google.com/drive/1-NqeOGQ6ZWlzuqx_VSHasY-IFhgnIWVt?usp=sharing`

2 Video - GPT: from scratch, in code, spelled out.

Durante o vídeo o autor revela as táticas usadas pelo chatGPT na sua implementação do núcleo. Para exemplificar com um exemplo prático ele resolve demonstrar como criar um "mini chatGPT de Shakespeare", fazendo com que escreva sequencialmente caracteres (tokens, uma vez que os tokens dessa arquitetura feita por ele são de tamanhos únicos de caracteres) simulando um texto escrito pelo autor Inglês.

Primeiramente ele separa o text em "Chunks"para ser usado durante o treinamento, e utiliza o modelo de linguagem de Bigram e após aplica a técnica de "self-attention", a parte principal do algoritmo, o qual consegue olhar para os tokens anteriores e gerar alguma conclusão, no caso, o próximo token. Também inclui no modelo, "dropout", conexões residuais de erros, e normalização de camadas.

Link Colab: `https://colab.research.google.com/drive/112yFuaNTqbqI4zG6i5GDYOC4HBhnQ8Or?usp=sharing`