Universidade do Minho
Escola de Engenharia

Autoencoders

Tecnologias e Aplicações

Trabalho realizado por:

João Linhares (A86618)

Junho de 2021

Índice

Lista de Figuras

Capítulo 1

Introdução

1.1 Contextualização

Na sequência da Unidade Curricular de Tecnologias e Aplicações do perfil de Computação Gráfica foi proposta a realização do presente projeto, cujo objetivo consiste em explorar modelos generativos. Para este trabalho foram utilizados *autoencoders* para criar estes modelos que por sua vez foram utilizados dois *datasets* para os treinar, aos quais irão ser referidos nos seguintes capítulos. Para além disso, foram explorados também *autoencoders* para finalidades de reconstrução, remoção de ruído e detecção de anomalias pois devido a infelizmente não ter podido estar presente na primeira aula de *autoencoders* achei bastante interessante explorar os mesmos.

Com isto o objectivo deste projecto é experimentar e visualizar os resultados obtidos em *autoencoders* com diferentes finalidades e casos de uso.

1.2 Estrutura do relatório

Este relatório está dividido em quatro partes distintas:

Esta primeira parte, corresponde à introdução, à qual é feita uma contextualização do projeto e é explicada a estruturação do relatório.

Na segunda parte 2 do relatório são indicados os *datasets* utilizados e para que finalidades estes foram usados.

Na terceira parte 3 do relatório são explicados os *autoencoders* criados e são mostrados os resultados obtidos.

A quarta e última parte 4 faz uma breve conclusão do trabalho elaborado.

Capítulo 2

Datasets

Neste projeto foram utilizados três *datasets* para constuir os vários *autoencoders* que irão ser mostrados ao longo deste relatório.

O primeiro foi o *dataset Fashion MNIST* que é um *dataset* bastante popular e muito utilizado para validar algoritmos. Este *dataset* pode ser obtido diretamente da api *Keras* do *Tenserflow* em que pode ser visto como foi carregado nos *notebooks* aos quais utilizam este *dataset*. Este é composto por 60000 imagens de treino e 10000 imagens de teste todas 28x28 em escala de cinzento. O tema deste *dataset* como pelo nome indica é uma coleção de diferentes tipos de roupa que está dividido em 10 classes sendo estas T-shirt, Calças, Pullover, Vestido, Casaco, Sandálias, Camisola, Sneakers, Mochila e Botas.

Figura 2.1: *Dataset MNIST Fashion*

O segundo *dataset* é um *Cartoon Set* que é uma coleção de imagens de avatar em 2D. Este pode ser obtido como um *set* de 10 mil ou 100 mil imagens através do link `https://google.github.io/cartoonset/index.html`. Para este projeto, este *dataset* foi aquele em que foi gasto mais recursos pois é um exemplo mais complexo. Foi utilizado o *set* de 100 mil imagens a cor de 256x256 pixeis para criar um *autoencoder* de reconstrução e um generativo aos quais podem ser vistos no próximo capítulo.

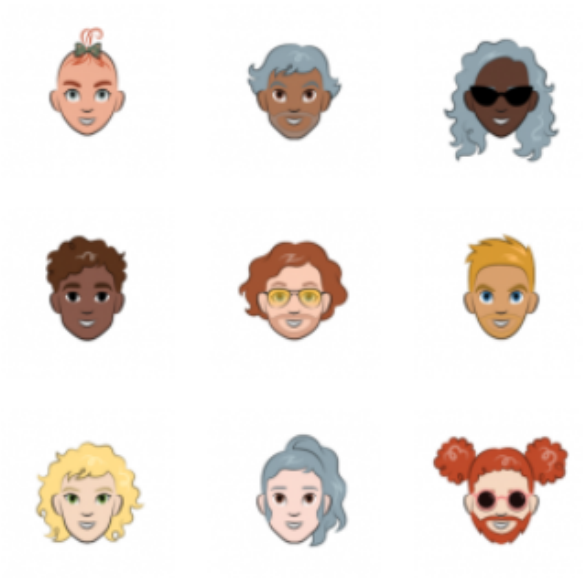

Figura 2.2: *Cartoon Dataset*

Por último, o terceiro *dataset* é um conjunto de dados de 5000 eletrocardiogramas que pode ser obtido através do link `http://www.timeseriesclassification.com/description.php?Dataset=ECG5000` ou através do download direto que pode ver visto no *Notebook 'TI_Autoencoders_Eletrocardiograma.ipynb'*. Este conjunto foi utilizado apenas para a secção 3.3 portanto foi utilizada uma versão mais simplificada, onde cada eletrocardiograma é rotulado com 0 se for um ritmo anormal e com 1 se for um ritmo normal.

Capítulo 3

Autoencoders

Neste capítulo irão ser explicados os vários *autoencoders* criados e os seus respectivos resultados.

Como dito anteriormente, foram explorados *autoencoders* para várias finalidades e casos de uso, sendo estes de reconstrução, remoção de ruído, deteção de anomalias e de geração.

3.1 *Autoencoders* de Reconstrução

Foram criados dois *autoencoders* com a finalidade de reconstrução, um para o *dataset* da roupa e outro para o mais complexo de *cartoons*.

3.1.1 *Dataset Fashion*

Este é um *autoencoder* básico com o objectivo de reconstruir imagens ao qual aprende a compactar as mesmas enquanto minimiza o erro de reconstrução.

Este foi treinado com 25 épocas a partir das imagens de treino do *dataset* do *Fashion MNIST*. A sua arquitetura é composta por um *encoder* de 9 *layers* e um *decoder* de 11 *layers* que podem ser vistos com mais detalhe no *Notebook 'TI_Autoencoders_Fashion.ipynb'*.

Como se pode reparar na figura 3.1 o mesmo consegue reconstruir as imagens originais dentro da mesma classe, ou seja, se uma imagem original é um par de calças este reconstroi um par de calças se for umas sapatilhas ele reconstroi um par de sapatilhas. Existem erros de reconstrução no entanto, por exemplo na linha 4, esta reconstrução não é perfeita apesar de ser na mesma classe existe diferenças significativas. Penso que se fosse aumentado o número de épocas iria haver melhorias significativas.

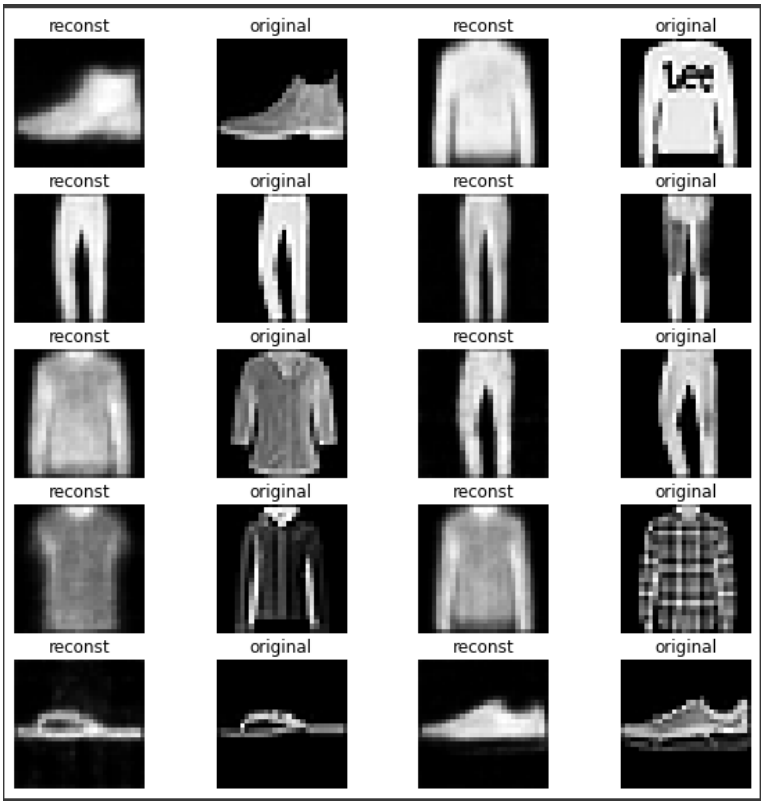

Figura 3.1: Resultados obtidos no *autoencoder Fashion MNIST*

3.1.2 *Dataset Cartoon*

Tal como na subsecção anterior, este é um *autoencoder* com o mesmo objectivo de reconstruir imagens. Neste caso, é aplicado sobre o *dataset* de *cartoons* que é bastante mais complexo e de maior dimensão.

Este foi treinado na mesma com 25 épocas e a sua arquitetura é composta por um *encoder* de 18 *layers* e um *decoder* de 16 *layers* o que representa uma rede de dimensão extremamente superior à anterior. Com isto devido aos fatores de complexidade do *dataset* e o facto da arquitectura ser mais complexa o tempo de aprendizagem foi extremamente superior. Como se pode ver no *Notebook 'TI- Autoencoders Cartoon.ipynb'* tanto o *encoder* como o *decoder* foram treinados separadamente para facilitar na criação de futuros testes sobre um dos dois e reconstruir imagens tanto de uma imagem original através de um *encoder* ou até criar vector de latentes e utilizar apenas o *decoder*. Esta técnica foi utilizada apenas neste *autoencoder* e nos da secção 3.4.

Através da figura 3.2 pode se reparar que apesar de ser um *dataset* mais complexo, o *autoencoder* fez um bom trabalho ao reconstruir as imagens, mesmo os detalhes mais finos são nítidos e bastante perceptíveis.

Figura 3.2: Resultados obtidos no *autoencoder Cartoon Set*

3.2 *Autoencoders* Para Remoção de Ruído

Este *autoencoder* tem como objetivo remover ruído das imagens do *dataset Fashion MNIST*.

O primeiro passo foi adicionar ruído às imagens da coleção para treinar o modelo obtendo o seguinte exemplo:

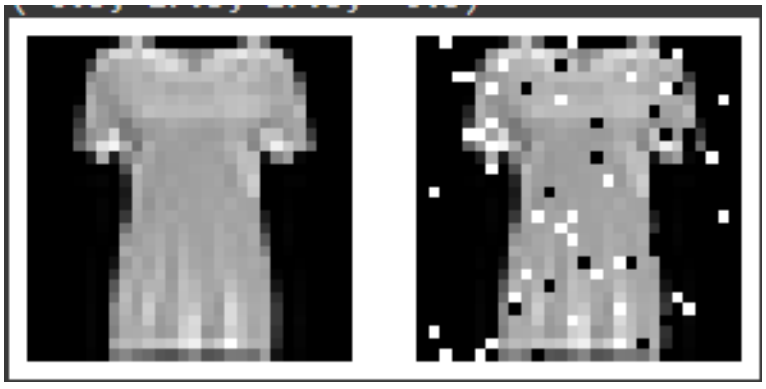

Figura 3.3: Exemplo de uma imagem com ruído

Devido a ser o mesmo *dataset* mas apenas com adição de ruído foi utilizada a mesma arquitectura à da subsecção 3.1.1 e de seguida o modelo foi treinado com 25 épocas com o respectivo conjunto de imagens com ruído.

Como se pode ver na figura 3.4 o ruído é removido com sucesso, tal como na subsecção 3.1.1 tanto o sucesso como os problemas são os mesmos, portanto um aumento no número de épocas poderia levar a um resultado melhor.

Figura 3.4: Resultados obtidos no *autoencoder* de remoção de ruído *Fashion MNIST*

3.3 *Autoencoders* de Deteção de Anomalias

Neste exemplo, irá ser treinado um *autoencoder* para detectar um electrocardiograma com anomalias de um conjunto de electrocardiogramas. Este é um caso de uso bastante interessante ao qual aproveita o facto de um codificador automático ser treinado para minimizar o erro de reconstrução para detectar os que têm anomalias dos que não têm.

Este modelo foi criado com uma arquitectura simples com um *encoder* e um *decoder* com apenas 3 *layers* devido à simplicidade do *dataset*.

Para detectar anomalias, o modelo foi treinado apenas com o conjunto de electrocardiogramas normais para depois considerar um eletrocardiograma como anômalo caso o erro de reconstrução for maior que um valor fixo. Este valor fixo foi calculado através do erro médio dos exemplos normais.

Como se pode ver no conjunto de figuras 3.7 e 3.8 é notável a distinção no valor de erro de reconstrução obtido no conjunto de ecgs normais dos anormais em que nos posteriores o erro de reconstrução é significativamente maior o que nos permite deduzir caso o erro de reconstrução de electrocardiograma for superior a um *threshold* podemos considerar o mesmo como anómalo.

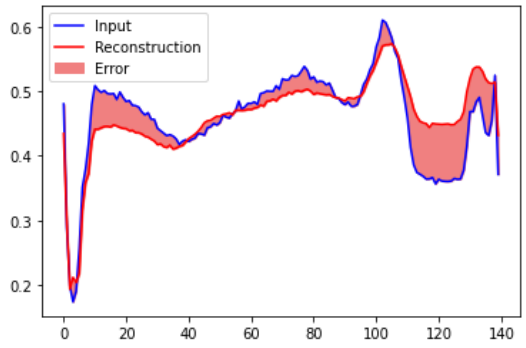

Figura 3.5: Electrocardiograma Normal

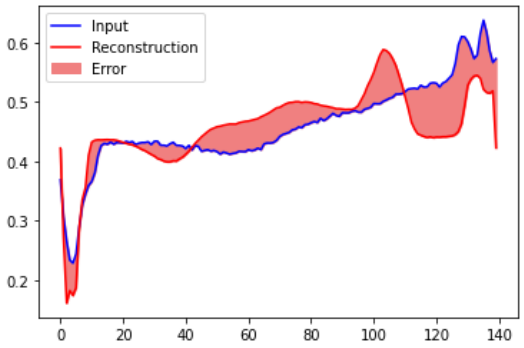

Figura 3.6: Electrocardiograma Anormal

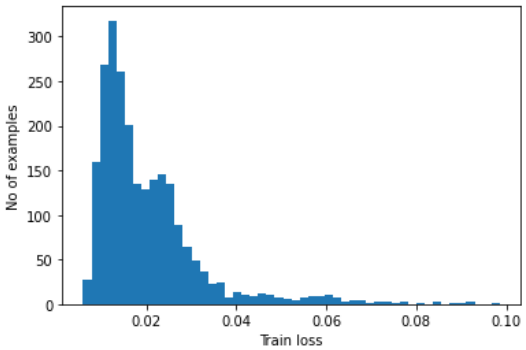

Figura 3.7: Erro de Reconstrução dos Electrocardiogramas Normais

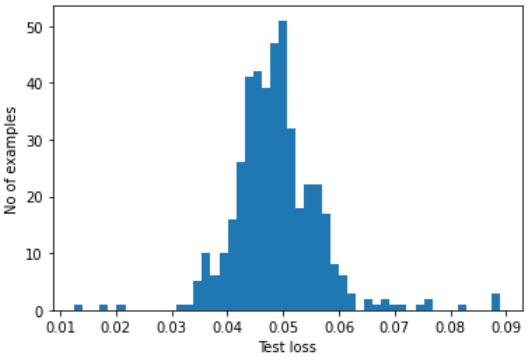

Figura 3.8: Erro de Reconstrução dos Electrocardiogramas Anormais

3.4 *Autoencoders* Generativos

Foram criados dois *autoencoders* generativos, um para o *dataset* da roupa e outro para o mais complexo de *cartoons*.

3.4.1 *Dataset Fashion*

Este é um *autoencoder* generativo treinado com 25 épocas com o objectivo de gerar novas imagens relacionadas com o *dataset Fashion MNIST* e utilizar o facto de ser um *dataset* mais pequeno para testar o algoritmo para ser utilizado na próxima subsecção que corresponde a um *dataset* mais complexo.

Os resultados obtidos na figura 3.9 foram obtidos a partir do *dataset* de teste, aos quais pode se reparar que o *autoencoder* conseguiu gerar imagens na mesma classe relativamente bem. Analisando as várias linhas percebe-se que quando a classe é um par de calças a imagem gera um novo par de calças. No entanto, existe certas imagens que apesar de parecer gerar a classe correcta, existem artefactos ou misturas com outras classes, por exemplo na linha 2 na penúltima coluna, apesar de gerar uma sandália é possível ver um par de calças atrás da mesma meio transparente.

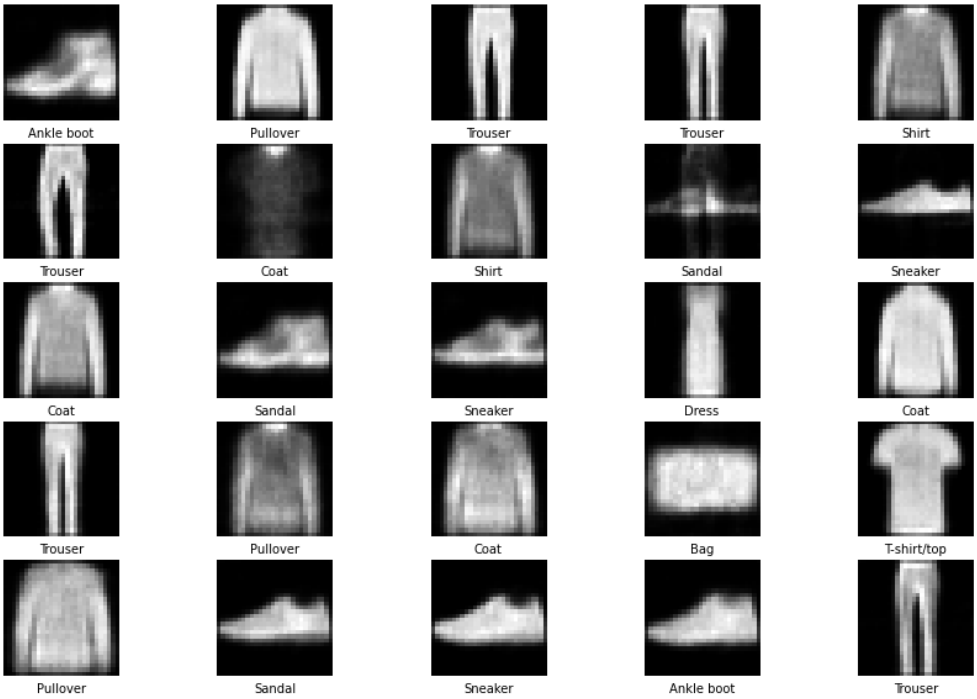

Figura 3.9: Resultados obtidos no *autoencoder* generativo *Fashion MNIST*

3.4.2 *Dataset Cartoon*

Este é um *autoencoder* generativo com o objectivo de gerar novas imagens relacionadas com o *dataset* de *cartoons*. Foi utilizado o mesmo algoritmo que na subsecção anterior, apenas o *encoder* e o *decoder* foram adaptados para aprenderem este *dataset* mais complexo com 25 épocas.

As figuras 3.10 e 3.11 representam os resultados obtidos através do *dataset* e através de um vector de latentes *sampled* com distribuição normal respectivamente. Após olhar para os resultados, pode se reparar que gera imagens nítidas e bastante perceptíveis em ambos os casos. É engraçado o facto do *autoencoder* na maioria das vezes aplicar barba a quase todos os *cartoons* gerados. Também, devido que no *dataset* existem bastantes *cartoons* com óculos de sol, em certas imagens geradas estes ficam meio transparentes parecendo olheiras como por exemplo as duas primeiras imagens na terceira linha da figura 3.10. É de notar também que em poucos dos casos existem certas falhas nos casos do cabelo comprido em que certas partes do cabelo desaparecem completamente. Do resto, tendo em conta a dimensão do *dataset* e do tamanho de cada imagem, com apenas 25 épocas obteve-se bons resultados.

Figura 3.10: Resultados obtidos no *autoencoder* generativo através de imagens do *dataset*

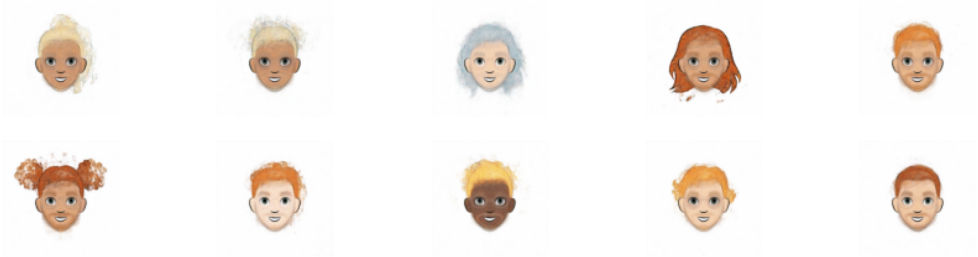

Figura 3.11: Resultados obtidos no *autoencoder* generativo através de um vector de latentes *sampled* com distribuição normal

3.5 Considerações finais

Todos os modelos têm os seus pesos na respectiva pasta dos respetivos notebooks. Também existem exemplos e funções auxiliares declaradas que permitem facilmente carregar estes pesos e avaliá-los, assumindo que os modelos estão nas pastas corretas, bem como o *dataset*. Para ver dados como o histórico de training, ou então exemplos de resultados, basta abrir os respetivos notebooks. Todos os *Notebooks* e ficheiros para fazer load dos modelos foram fornecidos com o relatório aos quais estão divididos entre pastas para ficar mais organizado. Estes também podem ser obtidos no repositório público do *github* através do link `https://github.com/JoaoLinhares/projecto-ta-individual`.

Capítulo 4

Conclusão

Em conclusão, apesar dos limites de recursos tanto como problemas a correr na minha máquina local utilizando o GPU e problemas nos limites definidos na utilização de recursos de aceleração no *Google Colab*, para cada *autoencoder* treinado foram obtidos bons resultados e o objectivo de visualizar praticamente diferentes *autoencoders* com finalidades e casos de uso diferentes foi cumprido com sucesso.

Com este trabalho aprendi que podem se fazer coisas bastante interessantes e até mesmo rapidamente com o uso de *autoencoders*, como por exemplo o *autoencoder* para detectar anomalias em electrocardiogramas e apesar de mais demorado mas com resultados mais interessantes a capacidade de gerar novas imagens.

Para trabalho futuro existem agora vários caminhos a seguir, um destes seria criar um *autoencoder* com um *dataset* ainda mais complexo talvez com imagens do dobro da resolução (512x512) e verificar os resultados obtidos. E também devido a explorar apenas *autoencoders*, explorar também o mundo dos *GANs*.

Bibliografia

Google. “Cartoon Set”. https://google.github.io/cartoonset/.

Physionet. “Dataset: ECG5000”. http://www.timeseriesclassification.com/description.php?Dataset=ECG5000.

Research, Z. 2017. “Fashion MNIST”. https://www.kaggle.com/zalando-research/fashionmnist.

“Tenserflow”. https://www.tensorflow.org/?hl=pt-br.