Relatório Deep Learning
Assignment 5: Deep k-Means

Ana Carolina Erthal & Felipe Lamarca

Link para o Google Colab

1 Modificações no código

Para esse assignment, as modificações no código ocorreram em menor escala quando comparamos a assignments anteriores. Como o código já era bastante completo para a primeira parte, focamos nossos esforços em compreender mais a fundo os conceitos necessários de Annealing, Temperature Scaling e principalmente da estrutura da Deep K-Means.

Seguindo o código, após importar os dados, definir toda a estrutura da rede e hiperparâmetros (expostos na próxima seção), realizamos os treinamentos determinados, variando entre a versão pré-treinada e a versão utilizando annealing, variando o α conforme pedido para cada situação. Realizamos, também, o cálculo das métricas de avaliação para todas as configurações, que serão discutidas mais adiante.

Por fim, realizamos a variação do números total de clusters para o melhor modelo obtido, buscando visualizar a performance deste modelo através da clusterização em mais ou menos clusters que o número total de classes.

2 Hiperparâmetros

Para todos os resultados apresentados abaixo, utilizamos os seguintes hiperparâmetros:

Hiperparâmetro	**Valor**
`n_epochs_AE`	50
`optimizer`	Adam
`learning rate`	$1e^{-3}$
`n_clusters`	2, 5, 10, 20
α_0 (α-Annealing)	0.01, 0.1, 0.5, 1
α_n (α-Pretrained)	3, 30, 150, 300
`# alphas`	40 (α-Annealing), 20 (α-Pretrained)

Tabela 1 – Hiperparâmetros

3 Resultados

3.1 Task 1

A tabela abaixo apresenta a acurácia dos modelos:

$\alpha-$**Annealing**	α**-Pretrained**	**Acurácia**
$\alpha_0 = 0.01$		27%
$\alpha_0 = 0.1$		78%
$\alpha_0 = 0.5$		39%
$\alpha_0 = 1$		11%
	$\alpha_n = 3$	88%
	$\alpha_n = 30$	80%
	$\alpha_n = 150$	85%
	$\alpha_n = 300$	90%

Tabela 2 – Acurácia dos modelos Deep k-Means com abordagens Pre-trained e Annealing

Antes de discutirmos cada configuração de modelo avaliando a matriz de confusão e distribuição de F1 scores, é interessante observar atentamente a tabela de acurácias, já que ela nos mostra um comportamento bastante esperado por parte dos testes.

De início, o fator mais óbvio da tabela é a prevalência dos modelos pré-treinados, que obtiveram acurácias maiores e mais consistentes do que os modelos em que utilizamos a estratégia de annealing. Isso é bastante justificado pelo que vimos anteriormente na disciplina: realizar pré-treinamentos é, em geral, uma estratégia muito benéfica, já que realizamos aproveitamos treinamentos intensos e mais pesados do que conseguiríamos computar com mais facilidade, trazendo muito aprendizado ao modelo.

Vale perceber, ainda, que a variação determinada para o α parte de valores maiores, o que provavelmente é justificado pela maior confiança esperada pelo modelo. Como estamos tratando de uma inverse temperature, valores maiores de α determinam uma calibração em que a função Softmax passa a ser mais sharp, e o modelo passa a ter mais confiança em suas predições. Sendo assim, determinar um α inicial um pouco maior para o modelo pré-treinado, que certamente aprendeu mais e realiza predições mais assertivas, parece uma decisão orientada.

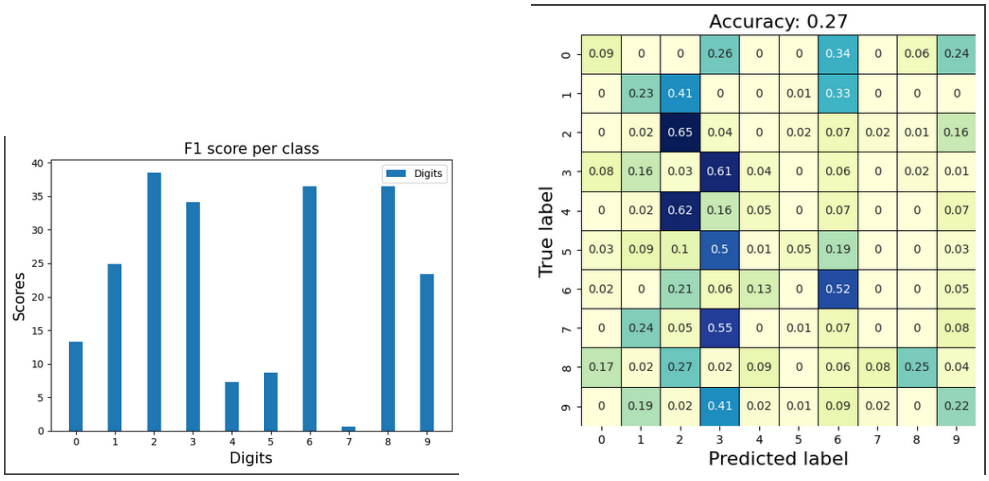
Analisando a diferença entre as acurácias para a versão annealing, também notamos um fator interessante! Nessa versão, observamos que a estratégia de exploração determinada pelo annealing, mas principalmente resultados provenientes da nossa escolha de α inicial. Como discutimos, para nossa inverse temperature, valores baixos representam uma calibração que torna a Softmax mais suave, tornando o modelo mais incerto, e valores altos tornam o modelo mais confiante. De fato, parece bastante razoável que o melhor modelo seja obtido quando partimos de um α nem tão alto nem tão baixo, e a partir do encontramos empíricamente um valor muito bom para partida.

3.1.1 Annealing

Vejamos agora resultados mais específicos para cada modelo e configuração.

Para o primeiro teste de annealing, quando partimos de um $\alpha = 0.01$, é possível observar a incerteza do modelo. Para cada classe, as predições são muito distribuídas, e contamos com pouca assertividade. Para o zero, por exemplo, obtivemos distribuição principalmente entre três clusters, com probabilidades muito próximos, e nenhum deles sendo o True Label.

Observando o F1 score, nota-se que obtivemos performance melhor para algumas classes e pior para outras, mas todos os valores são bastante baixos se comparados à performance de outras configurações do modelo.

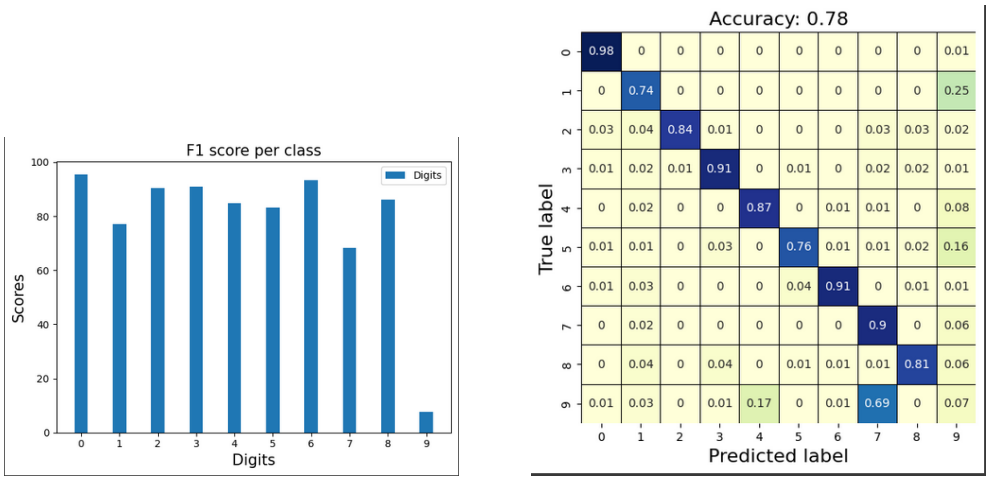


(a) F1 Score (b) Confusion Matrix

Figura 1 – F1 Score e Confusion Matrix para $\alpha_0 = 0.01$

Para o α inicial 0.1, no entanto, em que observamos a maior acurácia, a situação é muito diferente. O modelo é confiante na medida certa, e a calibração do Temperature Scaling parece ter sido bastante eficaz. Na matriz de confusão, obtemos uma ótima relação entre label predito e verdadeiro, com excessão do label 9, em que a performance foi ruim. Esse fato também pode ser observado no gráfico de F1 scores, em que todos os valores são bastante altos com excessão do score do dígito 9.

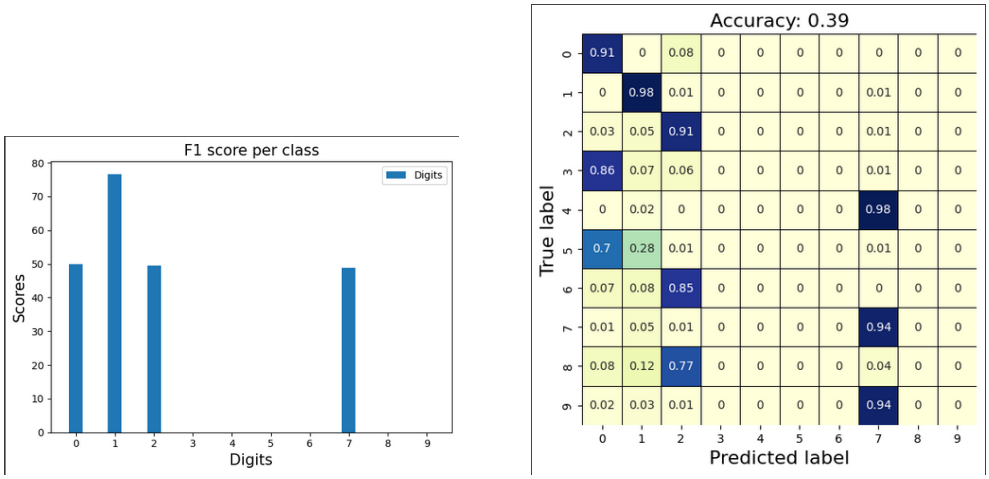
De fato, como mencionamos anteriormente, a busca pelo balanceamento entre incerteza e confiança obtido através da estratégia de annealing parece ter sido bastante eficaz nesse caso, em que encontramos um valor de α que trouxe um desempenho muito melhor ao modelo.



(a) F1 Score (b) Confusion Matrix

Figura 2 – F1 Score e Confusion Matrix para $\alpha_0 = 0.1$

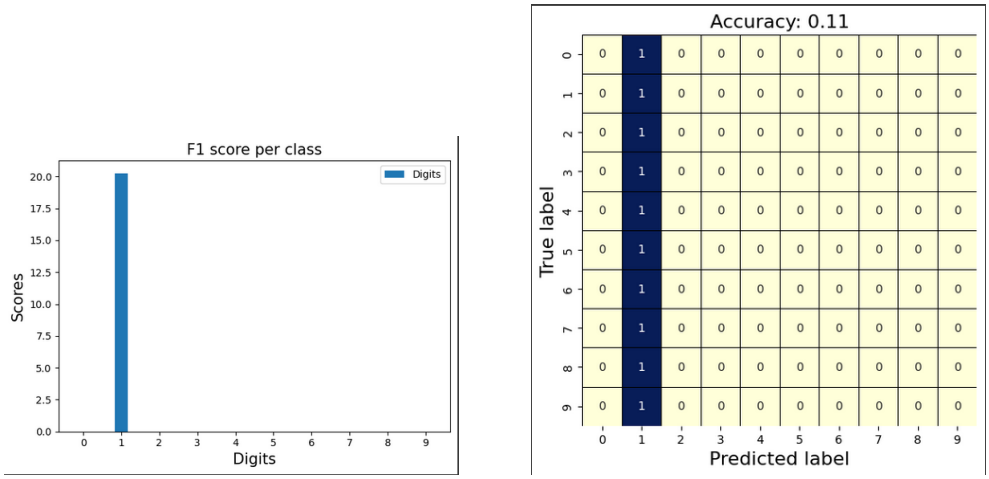
Para o $\alpha = 0.5$, começamos a visualizar o efeito do excesso de confiança do modelo causado pelo crescimento da inverse temperature. Temos uma Softmax sharp demais, e o modelo passa a atribuir predições com confiança excessiva, perdendo bastante em assertividade, como podemos observar nos dois gráficos. Note que há várias classes em que não há nenhum acerto.



(a) F1 Score (b) Confusion Matrix

Figura 3 – F1 Score e Confusion Matrix para $\alpha_0 = 0.5$

Por fim, temos o pior desempenho dentre os testes realizados, o modelo utilizando a estratégia annealing com um $alpha = 1$ de início. Aqui fica bastante claro o efeito de um excesso absoluto de confiança: o modelo passa a inserir todas as imagens em um mesmo cluster, tendo certeza (erradamente) de que todas as imagens pertencem a ele.

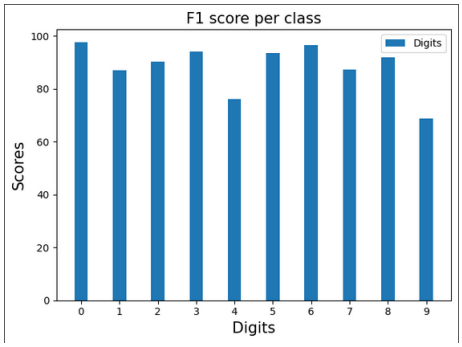


(a) F1 Score (b) Confusion Matrix

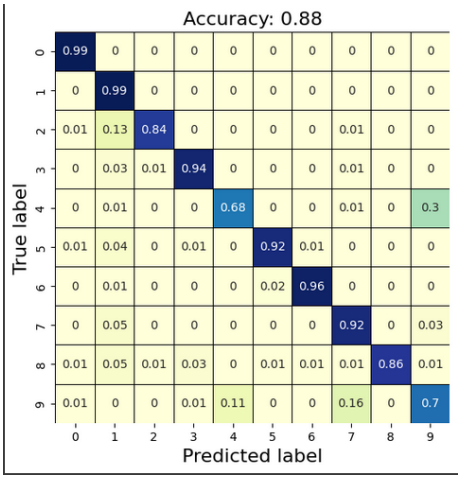
Figura 4 – F1 Score e Confusion Matrix para $\alpha_0 = 1$

3.1.2 Pretrained

No modelo pré-treinado, apesar de ser possível observar diferenças de acurácia e F1-scores, o valor inicial de α passa a ser menos significativo para o sucesso do modelo. Observe abaixo que todos os gráficos de F1 score produzidos, apresar de haver variações, se assemelham bastante, e se mantém em valores próximos (diferentemente do que observamos nos modelos com annealing).



(a) F1 Score

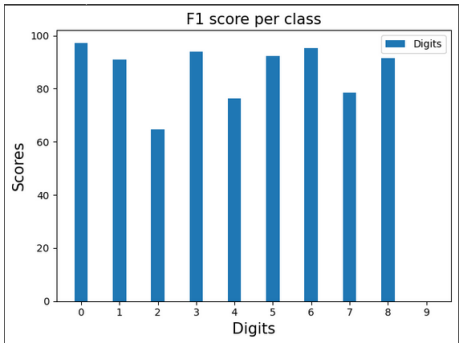


(b) Confusion Matrix

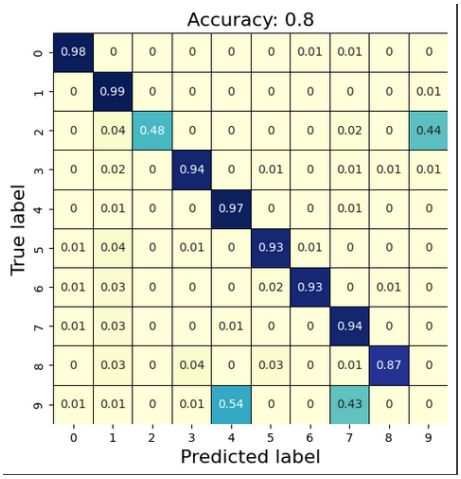
Figura 5 – F1 Score e Confusion Matrix para $\alpha_n = 3$

No primeiro caso, visto acima, partimos de um $\alpha = 3$, e obtemos um modelo bastante correto, com alguns erros maiores em classes específicas.

Note abaixo que quando variamos o α para 30, os erros passam a ser maiores, ocorrendo novamente a dificuldade de clusterizar imagens de label 9.

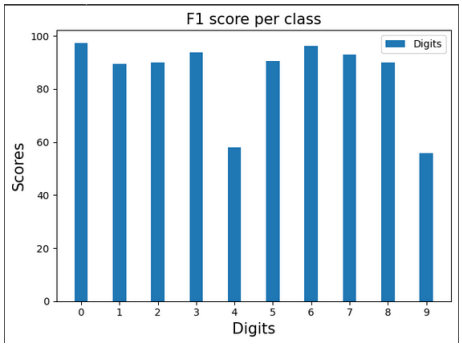


(a) F1 Score

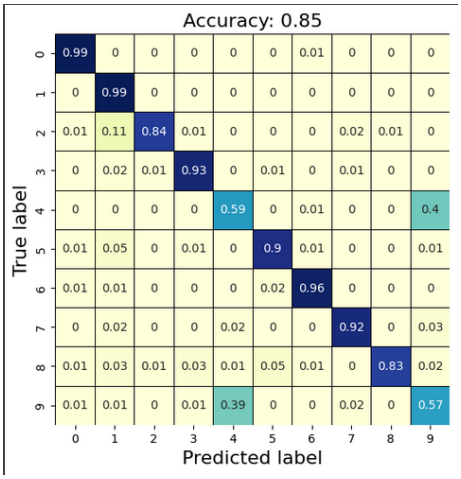


(b) Confusion Matrix

Figura 6 – F1 Score e Confusion Matrix para $\alpha_n = 30$



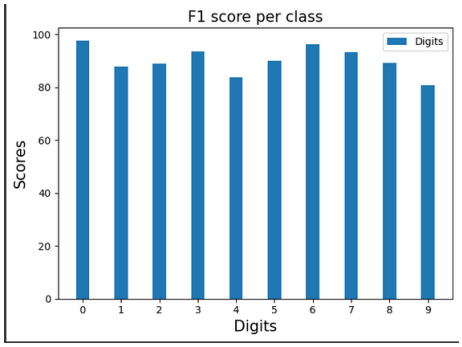
(a) F1 Score



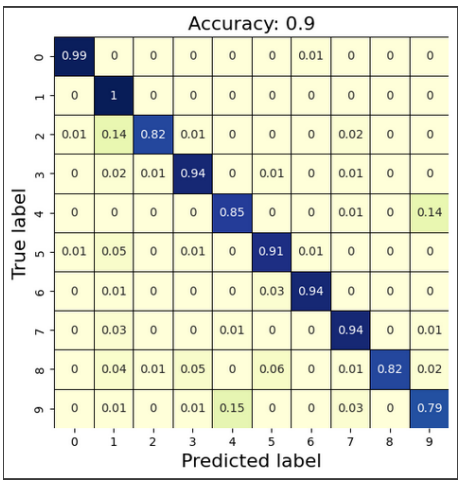
(b) Confusion Matrix

Figura 7 – F1 Score e Confusion Matrix para $\alpha_n = 150$

Quando determinamos $\alpha = 150$ seguimos obtendo um resultado próximo ao anterior: acertamos bastante, mas há incerteza em alguns labels, que são colocados em diferentes clusters com probabilidades próximas, demonstrando que ainda há melhoras a se realizar na calibração e, consequentemente, a escolha do α.



(a) F1 Score



(b) Confusion Matrix

Figura 8 – F1 Score e Confusion Matrix para $\alpha_n = 300$

Por fim, observamos nosso melhor resultado. Os labels previstos e verdadeiros são muito bem correlacionados, e todas as classes possuem um ótimo F1 score, demonstrando a eficácia da combinação entre valor inicial e do modelo pré treinado.

3.2 Task 2

Para essa task, variamos a quantidade de clusters existentes para observar a performance do melhor modelo (que, como vimos, foi o α-Pretrained com $\alpha_n = 300$) em

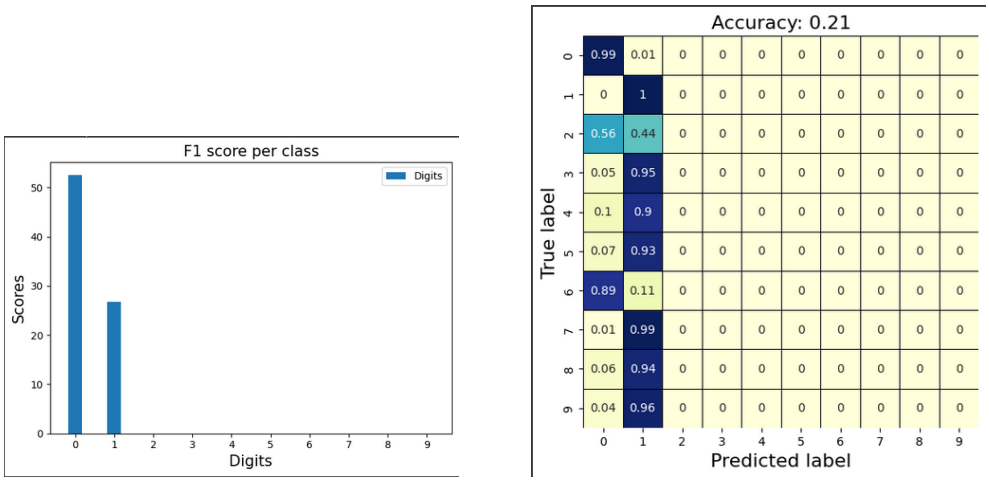
prever as classes. Veja abaixo a tabela de acurácias variando o número de clusters:

`n_clusters`	**Acurácia**
2	21%
5	49%
10	90%
20	55%

Tabela 3 – Acurácia dos modelos Deep k-Means com $\alpha_n = 300$

O melhor resultado é, com bastante distância para os demais, a situação em que temos 10 clusters, o que é bastante justificado pelo fato de que temos, na prática, 10 classes. A clusterização realizada pelo Deep K-means é obviamente mais correta ao predizer labels quando temos a mesma quantidade de classes e clusters, pois realizamos uma clusterização mais efetiva, buscando separar cada ocorrência de dígitos.

Veja abaixo, por exemplo, o resultado obtido para apenas 2 clusters. É claro que, quando há menos clusters que classes, nem mesmo chegamos a realizar predições para várias classes, o que traz uma taxa de erro bem alta. Além disso, observando exemplos de imagens atribuídas a cada cluster, fica bastante clara a deficiência (muito esperada) do resultado. Parece haver padrões nos agrupamentos, mostrando a efetividade do modelo em clusterizar, já que números como 4 e 9, mais semelhantes, parecem estar em geral no mesmo cluster, mas certamente as predições são muito ruins, já que só há 2 clusters.



(a) F1 Score (b) Confusion Matrix

Figura 9 – F1 Score e Confusion Matrix para $\alpha_n = 300$, `n_clusters = 2`

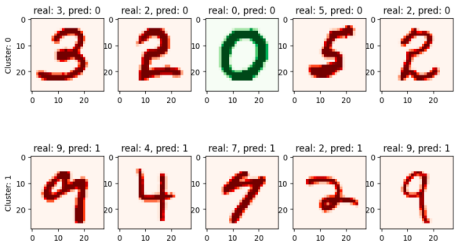
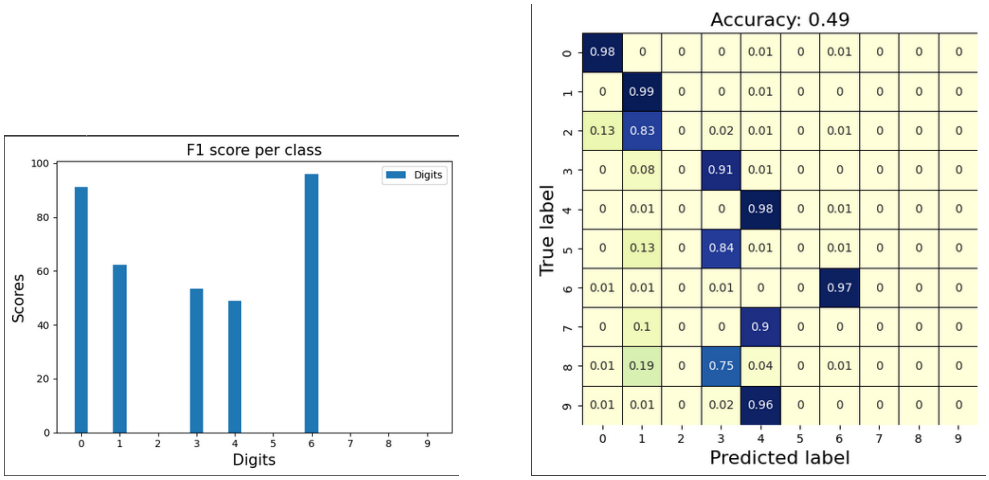


Figura 10 – Samples de imagens para `n_clusters = 2`

Com 5 clusters, temos um desempenho parecido com o resultado anterior, mas distribuindo nossas predições em, é claro, cinco classes diferentes. A acurácia aumenta correspondentemente, mas não há nenhuma predição para a outra metade das classes, resultando em uma acurácia ainda ruim. Pelas imagens, vemos que tivemos mais acertos, pois há um número maior de clusters (mais próximo ao ideal), mas ainda vários erros. Observe, novamente, o que destacamos anteriormente: há um padrão nesses agrupamentos, mostrando que realmente há uma clusterização efetiva para números mais próximos no espaço latente, ainda que a classificação seja deficitária pela quantidade de clusters.



(a) F1 Score (b) Confusion Matrix

Figura 11 – F1 Score e Confusion Matrix para $\alpha_n = 300$, `n_clusters = 5`

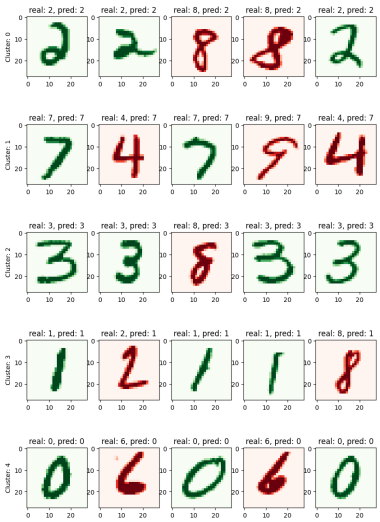
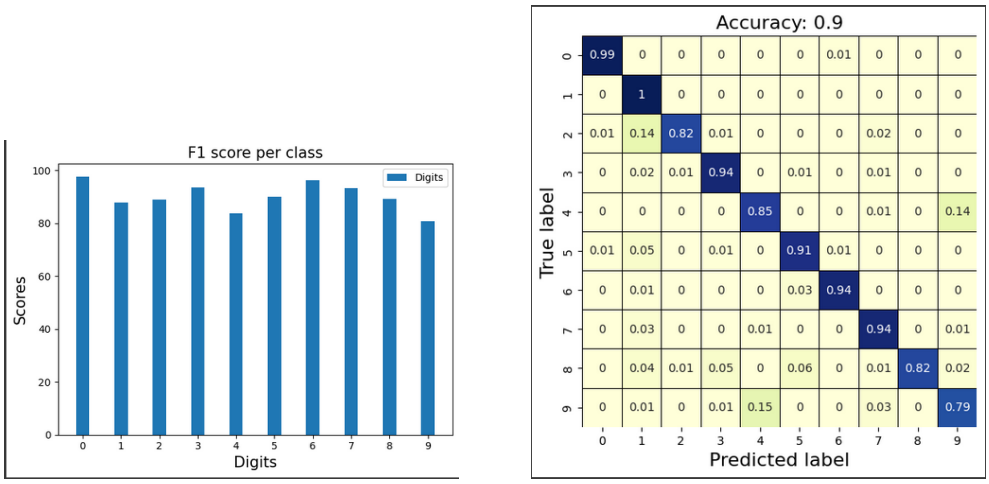


Figura 12 – Samples de imagens para `n_clusters = 5`

Veja abaixo o melhor resultado! De fato, como vimos anteriormente, tendo 10 clusters atingimos o melhor resultado. Os F1 scores são bastante altos, e a matriz de confusão demonstra a corretude das avaliações.

Nas imagens, visualizamos também esse sucesso. Com a quantidade correta de clusters, a acurácia se mostrou muito satisfatória, e as predições se mostram muito efetivas.



(a) F1 Score (b) Confusion Matrix

Figura 13 – F1 Score e Confusion Matrix para $\alpha_n = 300$, `n_clusters = 10`

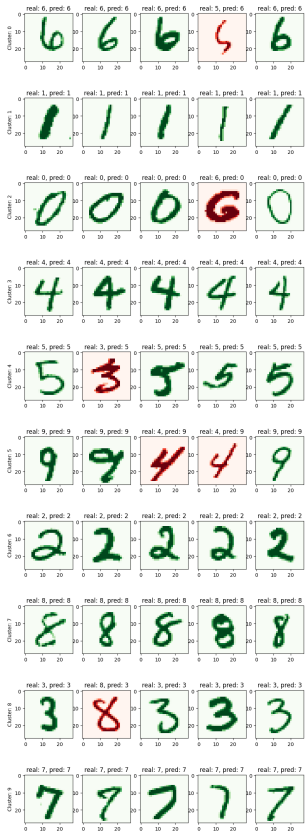
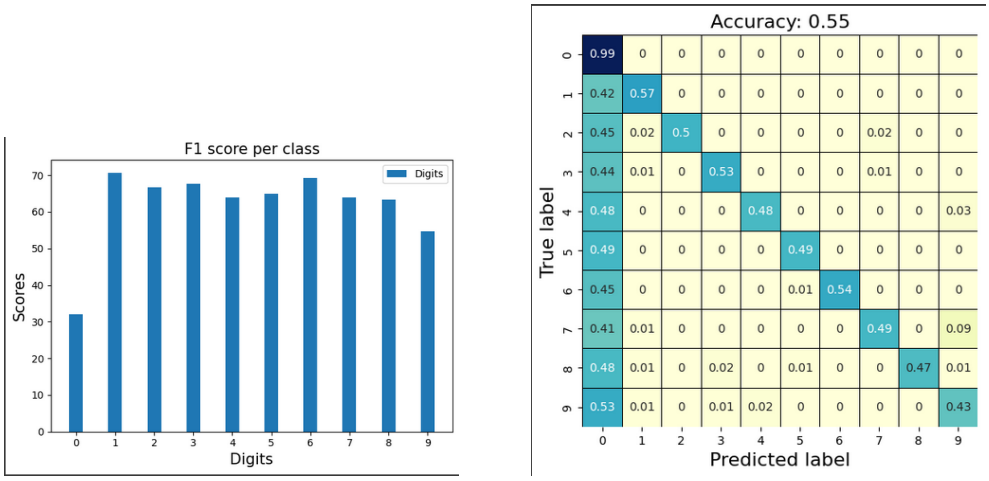


Figura 14 – Samples de imagens para `n_clusters = 10`

Por fim, quando temos 20 clusters (veja abaixo), observamos um fenômeno bastante interessante. As predições se distribuíram, principalmente, entre o true label e o zero. Isto é, o modelo tinha bastante incerteza entre clusterizar uma imagem como seu true label ou como um 0, e as probabilidades ficaram muito bem distribuídas entre esses dois casos.

O que ocorreu, quando analisamos as imagens, foi que para cada número, acabamos criando dois clusters, sendo um deles classificado como o número correto e outro como zero, justificando a matriz de confusão.



(a) F1 Score (b) Confusion Matrix

Figura 15 – F1 Score e Confusion Matrix para $\alpha_n = 300$, `n_clusters = 20`

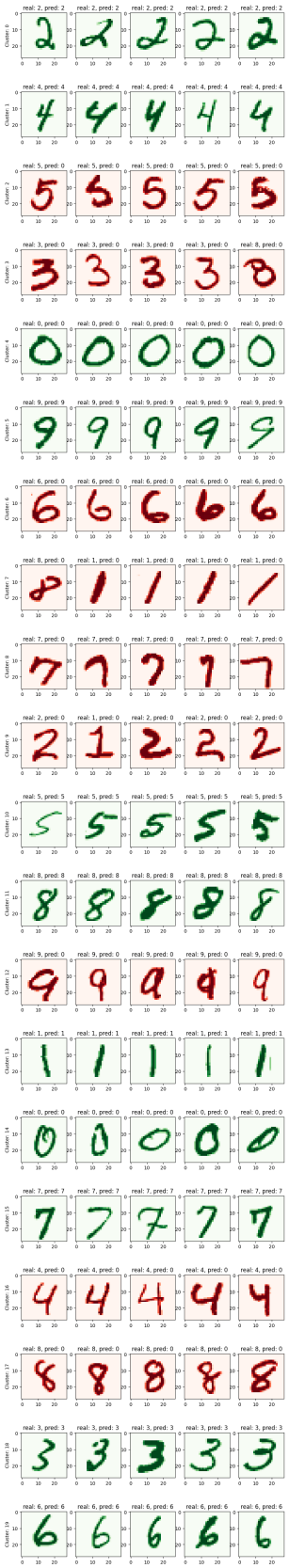


Figura 16 – Samples de imagens para `n_clusters = 20`

4 Avaliação dos assignments

Em geral, gostamos bastante do andamento e organização dos assignments. Um ponto bastante positivo foi o fato de o código já ser bastante avançado no notebook, o que nos permitiu gastar mais tempo entendendo o funcionamento dos modelos e resultados, e menos tempo debugando códigos e etc.. Também gostamos muito de receber os critérios de correção bem explícitos do monitor!

Como ponto negativo, teríamos sido bom se os códigos tivessem mais comentários. Em alguns assignments mais como esse, o código era bem grande e complicado, e foi mais difícil entender o que acontecia na implementação em algumas partes, o que teria sido interessante já que não estávamos implementando.