

Relatório Trabalho Prático 2: Reconhecimento de Captcha

CHAVES, Diogo Tuler

Aluno de graduação em Ciência da Computação da Universidade Federal de Minas Gerais

CAMPOS, João Marcos Tomaz Silva

Aluno de graduação em Ciência da Computação da Universidade Federal de Minas Gerais

1. Introdução

O propósito deste trabalho consistiu na implementação de dois métodos para o reconhecimento de cada um dos seis caracteres de um captcha. O primeiro método empregou a técnica de HOG, utilizando diversos classificadores, enquanto o segundo método consistiu na implementação de algumas redes neurais convolucionais (CNN), visando otimizar a taxa de reconhecimento e precisão.

2. Hog

2.1. Pipeline

Para a implementação do HOG, inicialmente, é crucial realizar a segmentação dos dados pertinentes, seguida pela aplicação de amostragem densa e extração das características desejadas. Em seguida, os dados são organizados em um dataframe, visando facilitar a compreensão e possibilitar a realização de tratamentos necessários.

Posteriormente, submetemos os dados a uma série de modelos durante as fases de treino e validação. Esse processo permite ajustes nos parâmetros e a seleção da configuração mais adequada aos resultados almejados. Uma vez obtidos resultados significativos nas métricas, o modelo é aplicado ao conjunto de testes para uma avaliação final.

2.2. Implementação

Para iniciar a implementação, é essencial realizar a extração das características das features. Para alcançar esse objetivo, cada imagem de captcha é dividida em seis partes, A extração das características é conduzida em cada parte da imagem, que é convertida para tons de preto e branco. Aplica-se o cálculo dos gradientes, a execução de janelas para amostragem densa, a criação de histogramas e, por fim, a concatenação em um vetor de features. Após essa etapa, cada vetor é registrado em um data frame, onde cada linha representa uma label com as features distribuídas em 1153 colunas. Para garantir um aprendizado eficaz, é necessário tratar as labels. Cada label respectiva ao caractere é colocada e ,em seguida, é aplicado o processo de one hot encoding, que associa cada label possível a um número.

Após isso, os dados são tratados de maneira eficiente para melhor resultado. Após essa etapa, eles são submetidos a diversos modelos para avaliação, tais como KNN, SVM, MLP, Logistic Regression, Decision Tree e Random Forest. Após ajustar os parâmetros conforme necessário, o conjunto de teste é utilizado para que o modelo realize as previsões.

2.3. Resultados

2.3.1. Exemplos de saída

Escolhemos previsões com erro para uma melhor análise. Erros estão em caracteres difíceis de identificar até para um humano.

	
Previsto foi IMIR90 e o real é IM1R9O	Previsto foi X7WBFZ e o real é X7W8FZ

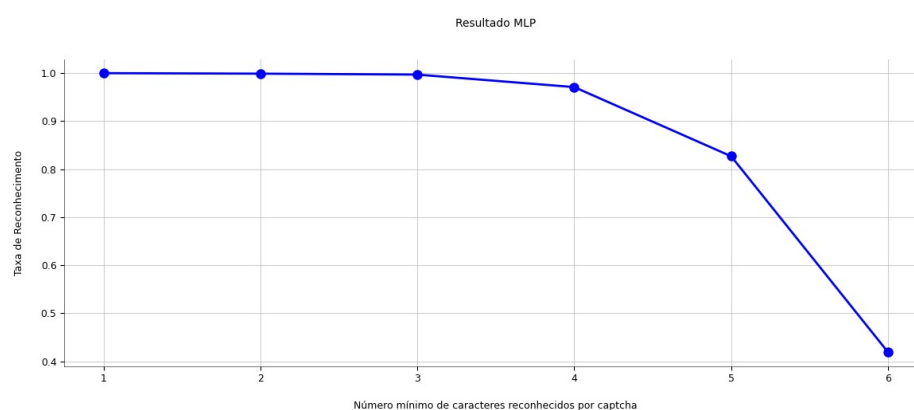
2.3.2. Acurácia dos modelos

Com a aplicação, os resultados variaram significativamente entre os modelos. À exceção de um modelo, todos os demais ficaram aquém dessa faixa de desempenho. Destaca-se que o modelo mais bem adaptado foi a MLP. Para sua implementação foi utilizado Adan de otimizador e a função loss de entropia cruzada.

	KNN	SVM	RF	RL	DT	MLP	SGDC
Acurácia	0.68	0.82	0.73	0.72	0.5	0.87	0.72

2.3.3. Melhor taxa de reconhecimento por caracter

Conforme previsto, a rede MLP alcançou a melhor taxa de reconhecimento para seis caracteres, uma vez que obteve a mais alta acurácia nos dados de teste para um único caractere.



3. CNN

3.1. Pipeline

Na tarefa de classificação de imagens por meio de uma CNN, é imperativo conduzir o pré-processamento de dados e estabelecer um carregador de dados para a modelagem. Após essa etapa, introduz-se um ciclo de treinamento e avaliação da função de perda nos conjuntos de treino e validação. Posteriormente, realiza-se a experimentação de diversos modelos, variando arquitetura e hiperparâmetros, com o intuito de otimizar a classificação dos dados. Por fim, os modelos finais selecionados são testados nos dados de teste, e suas métricas (incluindo recall, precisão, F1 e acurácia, no nosso caso) são comparadas para avaliação conclusiva.

3.2. Implementação

Para criar uma rede precisa de reconhecimento de caracteres em captchas, inicialmente, dividimos as imagens em seis segmentos iguais. Implementamos uma classe "Dataloader" para importar dados e converter classes em valores numéricos. Em seguida, desenvolvemos classes específicas para treino e teste do modelo.



Posteriormente, implementamos um modelo de CNN simples, com poucas camadas. Durante a fase de treino, ajustamos o modelo para encontrar a quantidade ideal de camadas, a estrutura da camada totalmente conectada e os hiperparâmetros ideais. Nosso modelo final possui duas camadas convolucionais intercaladas com a função de ativação "ReLU" e uma camada de "MaxPolling" seguidas de duas camadas totalmente conectadas com "Dropout" com a taxa de 0.4 e a função "ReLU" como função de ativação.

Finalmente, implementamos uma VGG16 (<https://arxiv.org/abs/1409.1556>), uma Resnet18 e uma Resnet34 (<https://arxiv.org/pdf/1512.03385.pdf>), seguindo os seus artigos originais com adaptações para o tamanho de nossos dados, para buscar acurácias maiores e possibilitar uma comparação com o modelo mais simples. Para todos os modelos o tamanho do batch foi de 64, e a função de perda foi a de entropia cruzada tendo em vista que seu bom desempenho em classificação de múltiplas classes é descrito na literatura, já a “learning rate” foi de 0.001 com o otimizador “Adam”.

3.3. Resultados

3.3.1. Exemplos de saída

Escolhemos previsões com erro para uma melhor análise. Erros estão em caracteres difíceis de identificar até para um humano.

	
Previsto foi JW1CZF e o real é JW4CZF	Previsto foi EMB4TN e o real é EM84TN

3.3.2. Acurácia dos modelos

Inicialmente, nossa expectativa era de que redes mais complexas, como as Resnet18 e 34, alcançariam acurácias superiores devido à sua maior profundidade. No entanto, os resultados obtidos surpreenderam, destacando a VGG como a rede com o desempenho mais elevado, aproximando-se significativamente da CNN simples, que, por sua vez, treinou de maneira consideravelmente mais rápida. Essa observação pode ser atribuída à simplicidade dos dados, compreendendo imagens simples com poucos pixels, indicando que um número reduzido de camadas já seria suficiente para a tarefa em questão.

	CNN Simples	VGG16	Resnet18	Resnet34
Acurácia	0.931	0.955	0.935	0.936

3.3.3. Melhor taxa de reconhecimento por caractere

Conforme previsto, a VGG alcançou a melhor taxa de reconhecimento para seis caracteres, uma vez que obteve a mais alta acurácia nos dados de teste para um único caractere.

