Projeto PSI3501: Verificação de Locutor

Piero Kauffmann

14 de Novembro de 2018

1 Introdução e Bibliográfia

Verificação de Locutor (VL) ou autenticação de locutor é uma tarefa de biometria que consiste em confirmar a identidade de um indivíduo i por meio de comparações entre uma nova instância de voz, obtida no momento da verificação (Y_i), e um banco de dados de gravações de voz (que possuí instâncias do usuário i assim como dos demais usuários do sistema).

Diferentemente dos sistemas de Identificação de Locutor (IL), sistemas de VL não procuram atribuir um locutor a um trecho de voz desconhecido, e sim autenticar a identidade alegada de um usuário. No contexto tradicional de biometria, espera-se que um sistema de VL opere com independência entre o algoritmo de verificação e o banco de dados de usuários cadastrados. Tal propriedade é essencial para o funcionamento do sistema pois permite que algoritmo de verificação se mantenha igual mesmo quando um usuário novo for registrado no banco de dados.

Sob a perspectiva de Aprendizado de Máquina, o problema de VL é geralmente visto como um problema de *Similarity Learning*, subárea de problemas cujo objetivo é estimar uma função de similaridade (ou de distância) entre instâncias que aproxime observações do mesmo usuário e ao mesmo tempo afaste observações de usuários diferentes.

Com os avanços recentes da pesquisa na área, algumas alternativas aos métodos tradicionais foram propostas. Em especial, destacamos Schroff et. al (2015) que propõe o aprendizado de uma função de distância que baseado em triplas compostas por: uma instância chave (ou *anchor*), uma instância positiva (de mesma identidade à instância chave) e uma instância negativa (de identidade diferente da instância chave). O problema reduz-se a encontrar uma função que diminua a distância entre a instância chave e a instância positiva e conjuntamente aumente a distância entre a instância chave e a instância negativa.

A proposta de aprendizado baseado em triplas é revisitado por Hermans et. al (2017), que propõe melhorias no algoritmo de treinamento por meio de *Hard Negatives Mining*, que consiste em escolher instâncias negativas que o modelo tenha dificuldade em reconhecer para compor as triplas.

Neste projeto, propomos implementar um sistema de VL baseado em Redes Neurais Convolucionadas baseado no modelo de verificação facial implementado por Schroff et. al (2015) e nas melhorias propostas por Hermans et. al (2017).

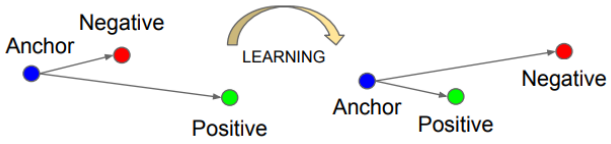

Figura 1: Ilustração do aprendizado usando *Triplet Loss*. Imagem extraída de Schroff et. al (2015).

2 Detalhes da proposta

2.1 Função de custo

O aprendizado baseado em triplas consiste em aprender os parâmetros θ de uma função de codificação $g_\theta : \mathbb{R}^N \to \mathbb{R}^D$ ($D \ll N$) que mapeie a j-ésima instância do usuário i, x_i^j, próximo às outras instâncias positivas $\{x_i^k, k \neq j\}$ e distante às instâncias de outros usuários $\{x_l^j, l \neq i\}$. Para fazer isso, uma instância chave do usuário i (x_i^a) é escolhida, assim como uma instância positiva (x_i^p) e negativa (x_j^n, $j \neq i$). Para cada tripla (x_i^a, x_i^p, x_j^n) a função de custo é calculada por

$$\mathcal{L}_{tri}(\theta) = \sum_{i \neq j} \sum_{\substack{a,p,n \\ a \neq p}} [m + \|g_\theta(x_i^a) - g_\theta(x_i^p)\|^2 - \|g_\theta(x_i^a) - g_\theta(x_j^n)\|^2]_+$$

Onde $[.]_+$ é a função $ReLU(x) = max(0, x)$ e m é um parâmetro de margem escolhido de antemão.

A dificuldade de implementação dessa função de custo vem do fato do número de triplas possíveis em função do tamanho N_{total} da amostra ser $O(N_{total}^3)$. Além disso, essa escolha de função de custo considera excessivamente os casos triviais em que x_i^a é muito diferente de x_i^n em detrimento dos casos mais importantes em que a dupla se confunde com maior facilidade.

Para mitigar esses dois problemas, Hermans et. al (2017) propõe uma função de custo alternativa (*Batch Hard*) que amostra *mini-batchs* compostos por M usuários da base de dados, onde, para cada um são sorteadas N instâncias, totalizando assim MN instâncias. Com isso, a função de custo do *mini-batch* é obtida por

$$\mathcal{L}_{BH}(\theta) = \sum_{i=1}^{M} \sum_{a=1}^{N} [\, m + \max_{p=1,2,\ldots N} \|g_\theta(x_i^a) - g_\theta(x_i^p)\|^2 - \min_{\substack{j=1,\ldots,M \\ n=1,\ldots,N \\ j \neq i}} \|g_\theta(x_i^a) - g_\theta(x_j^n)\|^2 \,]_+$$

Onde $\max_{p=1,2,\ldots N} \|g_\theta(x_i^a) - g_\theta(x_i^p)\|^2$ corresponde a maior distância possível entre a uma instância positiva e a instância chave (*Hardest Positive*) e o terceiro termo do somatório corresponde a menor distância possível entre uma instância negativa e a instância chave (*Hardest Negative*).

Vale notar que, devido ao procedimento ser baseado em amostragem, as triplas escolhidas para o cálculo da função de custo $\mathcal{L}_{BH}(\theta)$ podem ser vistas como triplas de dificuldade moderada, visto que as amostras obtidas de M usuários e N trechos nem sempre garantem a presença de *hard positives* ou *hard negatives*.

2.2 Especificação de g_θ

Para a função g_θ, é possível adotar tanto redes neurais convolucionadas (CNN) ou redes neurais recorrentes (RNN). As redes neurais convolucionadas utilizadas neste projeto, seguiram arquiteturas similares á rede neural VGG-16, assim como nas redes neurais com recorrência foram utilizadas arquiteturas baseadas em LSTMs bidirecionais.

3 Base de dados

3.1 MNIST

Figura 2: Exemplo de 70 imagens da base de dados MNIST.

A base de dados MNIST é uma base de dados famosa na área de visão computacional e é considerada um *benchmark* inicial para muitos tipos de problema.

A base consiste em 70 mil imagens de dígitos manuscritos por seres humanos, com os rótulos reais dos digitos que cada pessoa quis desenhar.

Para este projeto, usaremos a base de dados imaginando que cada dígito é um usuário (N = 10 digitos) com um determinado número de instâncias por usuário (M = 6 mil imagens).

3.2 VoxForge

Uma outra base de dados escolhida para o projeto foi uma base de dados coletada do repositório gratuito VoxForge [http://voxforge.org/], contendo gravações de conteúdo livre de até 12 segundos para 195 usuários (Ver apêndice, seção 5). Cada usuário em média apresenta 15 gravações (Figura 3). Os arquivos da base de áudio foram disponibilizados no formato *wav* com frequência de amostragem de $22050Hz$. Não existe restrição quanto ao uso de equipamentos específicos para a gravação de cada trecho de áudio.

A base atualmente está sob a licença GPL (GNU General Public License) e está disponível para uso livre.

Com a base de dados em questão, a tarefa de aprendizado associada é a de verificação de locutor do tipo *text independent*.

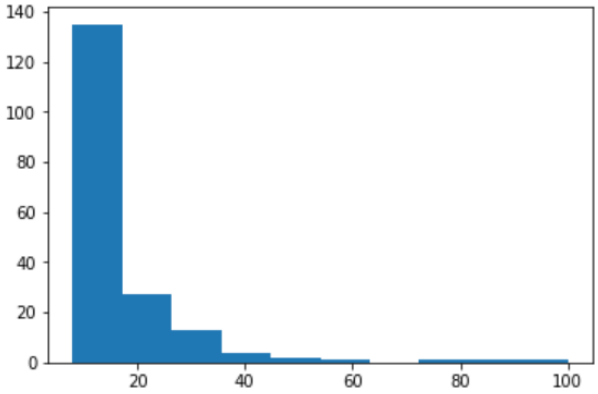

Figura 3: Distribuição do número de gravações de cada usuário da base VoxForge.

A etapa de pré-processamento e extração de *features* foi executada em três etapas:

- Padronização da duração de todos os arquivos de áudio para 10 segundos. Os arquivos com duração superior foram cortados após os primeiros 10 segundos e para os arquivos de duração inferior, foram adicionados pontos com amplitude nula para preencher os 10 segundos.
- Extração de 20 coeficientes Mel-Cepstrais (MFCC).
- Cálculo do Z-score de cada um dos coeficientes Mel-Cepstrais para que estes possuam média zero e desvio-padrão unitário.

4 Resultados

4.1 MNIST

Para a base de dados MNIST, o modelo utilizado foi uma rede neural convolucionada (CNN), com arquitetura descrita pela sequência:

- **Input**: Tensor (N x 28 x 28)
- Camada de 64 convoluções com kernel 3x3
- Camada de *max pooling* com kernel 2x2
- Camada de 128 convoluções com kernel 5x5
- Camada de *max pooling* com kernel 2x2
- Camada densa (*multi layer perceptron*) com 300 neurônios e função de ativação ReLU
- Camada densa (*multi layer perceptron*) com 300 neurônios e função de ativação identidade
- **Output**: Tensor (N x 300)
- Função de custo: *batch hard*, $\mathcal{L}_{BH}(\theta)$.

Após treinar o modelo, adotando-se N=6 dígitos e M=10 imagens, em 700 iterações, foram obtidos bons resultados. Uma matriz de distância calculada entre as projeções produzidas por g_θ segue representada na figura 4, lado direito. Observa-se que as observações de mesmo dígito possuem distância inferior (diagonal principal), como desejado. A função de custo (figura 4, lado direito) apresentou comportamento instável, porém decresceu satisfatóriamente após as 700 iterações.

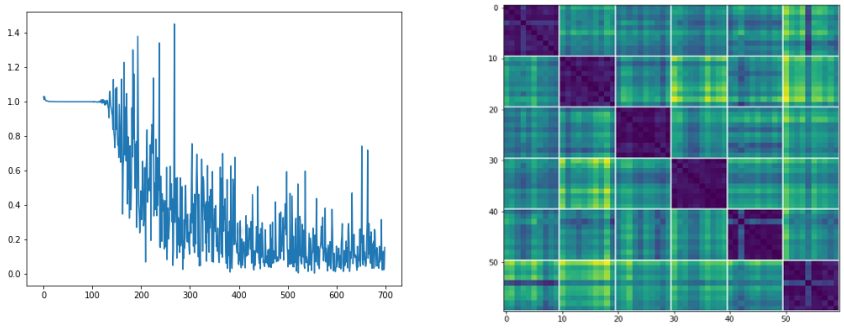

Figura 4: (Esquerda) Função de custo no treinamento. (Direita) Matriz de distâncias calculadas para $g_\theta(x_{ij})$ após 700 iterações de treinamento. Na figura, estão N=6 digitos, cada um representado por M=10 imagens.

Para verificar a qualidade dos resultados, aplicou-se uma técnica de redução de dimensionalidade nas dimensões obtidas pela aplicação de g_θ no conjunto de testes (Figura 5). Pode-se observar pela disposição dos agrupamentos que o modelo aprendeu a diferenciar os digitos de maneira correta.

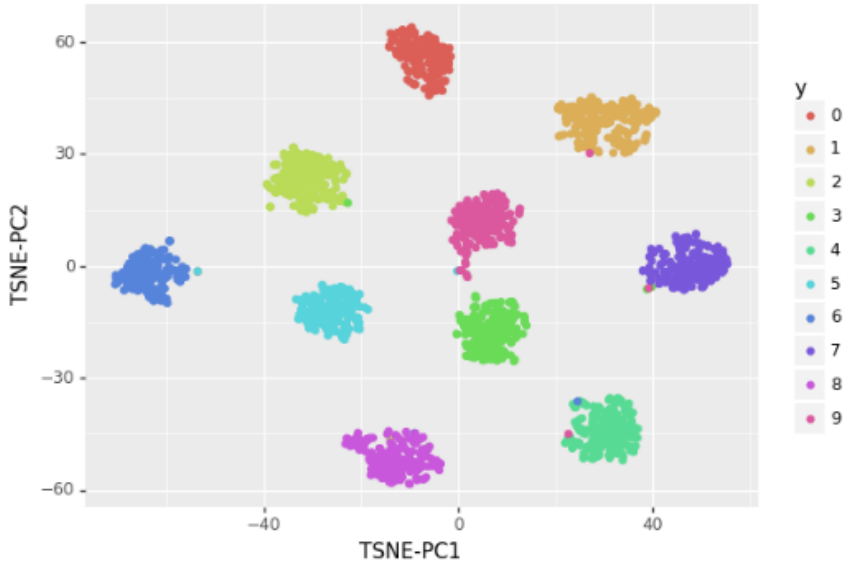

Figura 5: Pontos do conjunto de teste após a aplicação do algoritmo de redução de dimensionalidade T-SNE. Para cada ponto, estão indicadas o número verdadeiro do digito (cor no gráfico).

4.2 VoxForge

O modelo adotado para g_θ na base VoxForge foi uma LSTM bidirecional com arquitetura descrita na sequência:

- **Input**: Tensor N (batch) x 20 (features) x 431 (tempo)
- Camada LSTM com 16 *hidden units*
- Camada LSTM invertida com 16 *hidden units*
- Concatenação das duas últimas dimensões, produzindo como resultado um Tensor N X (32*431)
- Camada densa (*multi layer perceptron*) com 512 neurônios e função de ativação ReLU
- Camada densa (*multi layer perceptron*) com 300 neurônios e função de ativação identidade

- **Output**: Tensor N x 300
- Função de custo: *batch hard*, $\mathcal{L}_{BH}(\theta)$.

Para o treinamento do modelo, foram utilizadas amostras aleatórias 5 gravações para cada um de 10 usuários, totalizando 50 observações por iteração. O modelo adotado, apesar de apresentar tambem um comportamento instável, atingiu convergncia após 700 iterações (Figura 6).

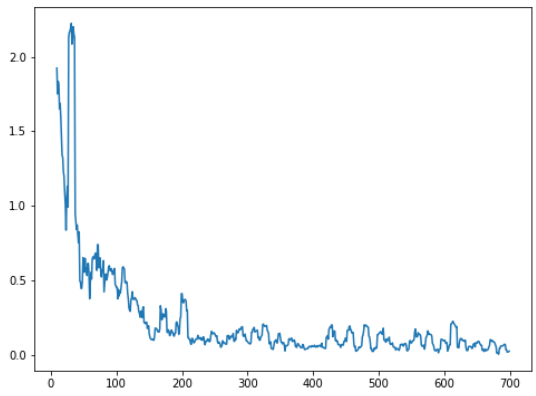

Figura 6: Média móvel (janela de 10 pontos) para a função de custo segundo o número de iterações.

Avaliação da qualidade do modelo

Assim como no modelo anterior, verificou-se, por meio de um método de dimensão de dimensionalidade (T-SNE), as projeções geradas pelo modelo para os dados de teste. Essas projeções seguem representadas na figura a seguir

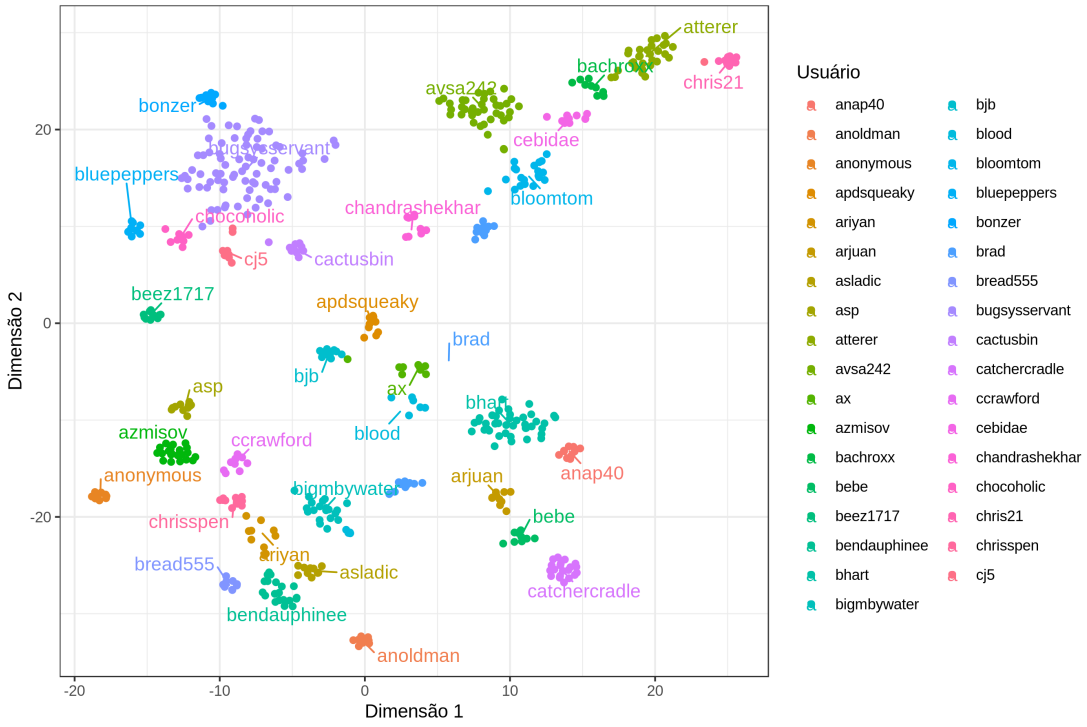

Figura 7: Projeções da função g_θ calculada para os dados de teste. Cada ponto representa uma gravação de voz, feita por um usuário.

Observa-se da Figura 7 que o modelo aprendeu a diferenciar gravações de vozes de pessoas desconhecidas, demonstrando uma boa capacidade de generalização. Da Figura 7, observa-se também a ocorrência de um caso atípico do usuário "brad", onde as gravações são separadas em dois subgrupos distantes entre si. Após uma verificação manual das gravações deste usuário, verificou-se a existência de áudios gravados por outra pessoa com o mesmo username.

Uma outra estratégia adotada para avaliar a qualidade do modelo é baseada no método dos K vizinhos mais próximos. A estratégia consiste em verificar para cada instância da base de testes, quais são os k pontos (ainda da base de testes) mais próximos. Caso a maioria destes k pontos pertencam ao mesmo usuário do ponto de referência, considera-se que o modelo acertou. Esse processo é repetido para a base inteira de testes e os acertos são contabilizados.

Os resultados produzidos pela aplicação dessa metodologia, representados

	K=1	K=3	K=5
Taxa de acertos (%)	100%	99,64%	99,74%

Tabela 1: Taxas de acerto do modelo para os K vizinhos mais próximos de cada ponto do conjunto de testes.

na Tabela 1, demonstram a boa eficiência do modelo para esta base de dados. Vale lembrar que a base de teste é composta por apenas 38 usuários, e que num contexto real este número seria muito maior e isso impactaria na performance do modelo.

5 Apêndice: Usuários da base VoxForge

5.1 Dados de treino

Lista de usuários do conjunto de treino com o respectivo número de amostras:

1snoke (10)
2old2play (10)
Aaron (20)
Adminvox (31)
AdrianMcNear (10)
Airwings (10)
AlexanderHenn (20)
Andrew (10)
AndreyBarri (10)
Angus (10)
Anniepoo (30)
Ara (10)
AslakKnutsen (10)
B (20)
Bahoke (40)
Berrym (10)
BlueAgent (9)
Bob (10)
Bossalicious (10)
Buddyboy (9)
CNJ (20)
Campbell (10)
Catbells (39)
CbS (10)
Chandri (10)
Chas (10)
CheesyBreed (30)
Chirag (9)
ColinBeckingham (90)
Coren (60)
Cristbal (10)
DanielHeath (20)
Daniil (10)
DantheDentalman (10)
Darrr (10)
DavidG (10)
DavidL (20)
Dcoetzee (50)
DeMilano (10)
Decent (20)
DeepBlue (10)
Donato (10)
Doogie (10)
DouglasDecicco (10)
ESimpray (10)
Edman274 (10)
Eresus (10)
Ertain (50)
Everett (10)
FatFrankie (10)
Flowerbot (20)
FrankdeLange (10)
FrozenFire (30)
GaylandGGump (30)
Gerard (10)
Gerry (10)
Gowrav (10)
Graham (10)
Gulain (10)
Hadlington (10)
JanusDaniels (10)
Jared (10)
Jay (10)
JayCutlersBrother (30)
Jennichan (10)
Joey (10)
John (10)
Joseph (10)
KalikiCEO (10)
KaloyanKolev (20)
Karageorgos (10)
Katzer (10)
KenBloom (20)
KenMacRaild (10)
Keren (10)
KnitGirl (10)
Krekos (10)
Krellis (20)
Kyton (30)
L1ttl3J1m (10)
LAVEENAZ (10)
LRMAAX (10)
Lexen (10)
Logomancer (20)
Luca (10)
Luis (10)
LuisManuelPr(10)
LukasFT (10)
LukeSkywalker (10)
LunaTick (10)
MJ (10)
MSypkensSmit (30)
Mariane (30)
Mike (20)
Nadim (20)
Paddy (20)
ParthC (10)
PaulB (10)
PaulWilliams (10)
Perygryne (10)
Peter (10)
PeterGarnett (20)
Peyrine (10)
PheonixK (10)
R (10)
RG (10)
Rain (30)
RedCisc (10)
Rishi (10)
RobSmith (20)
Ronin (10)

RonnelWing (10)
Ryan (20)
Saul (10)
Scott (10)
SergioDaroca (10)
Shane (10)
Shyam (10)
Sid (10)
Spacefish (10)
Splitlocked (10)
Steltek (10)
Sterna (20)
Sundar (10)
TLC (10)
Tamashii (10)
Tebbra (10)
TechnoZeus (30)
ThatsJames (10)
Tim018 (10)
TimS (20)
Timsage (10)
TomM (10)
Traktion (10)
Vineeth (10)
Vistaus (10)
Vortex (10)
Willem (9)
Wisleyv (10)
Zorthius (10)
adgar (8)
adubra (10)
aileen (10)
akiplaner (100)
alec (10)
alienation (8)
allison (20)
alphadecay (10)
amichaischreiber (10)
amolmg (31)

5.2 Dados de teste

Lista de usuários do conjunto de teste com o respectivo número de amostras:

anap40 (10)
anoldman (10)
anonymous (10)
apdsqueaky (10)
ariyan (10)
arjuan (10)
asladic (10)
asp (10)
atterer (30)
avsa242 (39)
ax (10)
azmisov (20)
bachroxx (10)
bebe (10)
beez1717 (10)
bendauphinee (20)
bhart (40)
bigmbywater (20)
bjb (10)
blood (10)
bloomtom (20)
bluepeppers (10)
bonzer (10)
brad (20)
bread555 (10)
bugsysservant (80)
cactusbin (10)
catchercradle (20)
ccrawford (10)
cebidae (10)
chandrashekhar (10)
chocoholic (9)
chris21 (10)
chrisspen (10)
cj5 (10)

6 Referências

HERMANS, A., BEYER, L., e LEIBE, B.; "**In Defense of the Triplet Loss for Person Re-Identification**".

SCHROFF F., KALENICHENKO D., e PHILBIN J.; "**FaceNet: A Unified Embedding for Face Recognition and Clustering**".

LECUN Y., BOTTOU Y., BENGIO Y., and HAFFNER P. **"Gradient-based learning applied to document recognition."** Proceedings of the IEEE, 86(11):2278-2324, November 1998. [on-line version]