unesp

Faculdade de Ciências

Uso de Redes Neurais Prototípicas em tarefas de classificação few-shot

Giovani Candido – Abril de 2021

Universidade Estadual Paulista “Júlio de Mesquita Filho” (UNESP)
Bauru, SP - Brasil

Tópicos a serem discutidos

1. Few-shot Learning (FSL)
2. Redes Prototípicas (RPs)
3. Variantes das RPs

Few-shot Learning (FSL)

O que é?

- É uma subárea de Machine Learning;
- Também conhecida por Low-shot ou n-shot learning;
- Lida com situações nas quais há poucos dados rotulados.

WANG, Y.; YAO, Q.; KWOK, J.; NI, L. M. Generalizing from a Few Examples. 2020.

Aspirações

- Aproximar ML e o aprendizado humano;
- Criar modelos que aprendam com poucos exemplos rotulados;
- Possibilitar que os modelos generalizem para novas tarefas;

LAKE, B. M.; SALAKHUTDINOV, R.; TENENBAUM, J. B. Human-level concept learning. 2015.

A

B

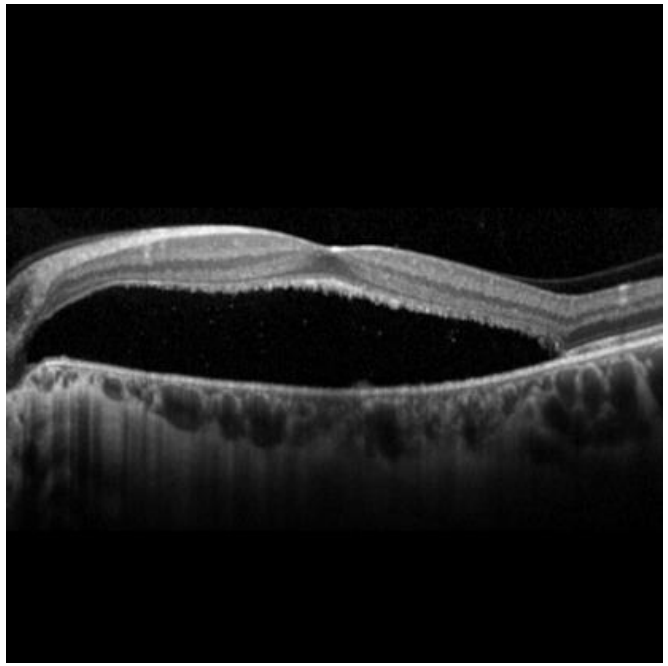
ಬು

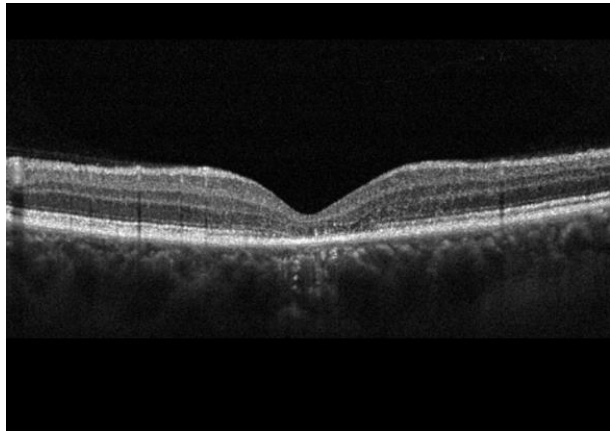
ಅ	ಇ	ಉ	ಎ	ಔ
ಕ	ಖ	ಗ	ಜ	ಝ
ಞ	ಠ	ಣ	ತ	ದ
ನ	ಯ	ಲ	ಹ	ಳ

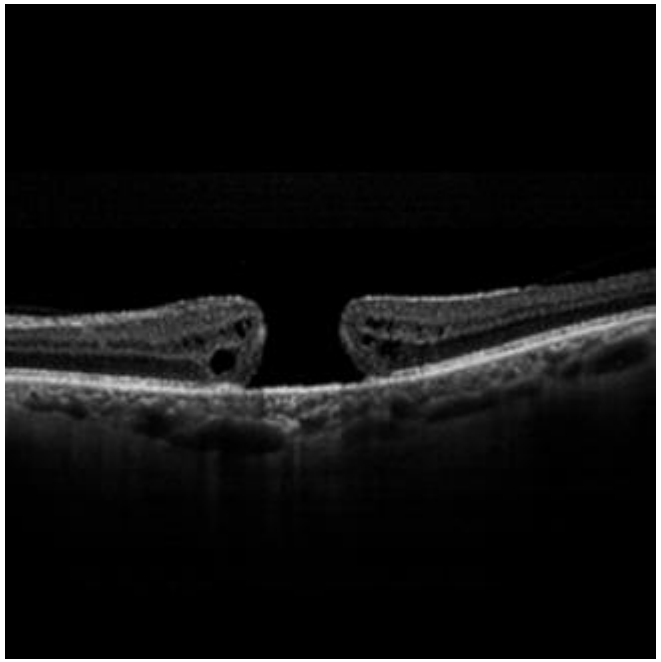
Aspirações

- Diminuir a coleta e o hand labeling de dados;
- Possibilitar modelos classificatórios de menor custo;
- Realizar tarefas de classificação com elementos raros.

Retinal OCT images of rare diseases



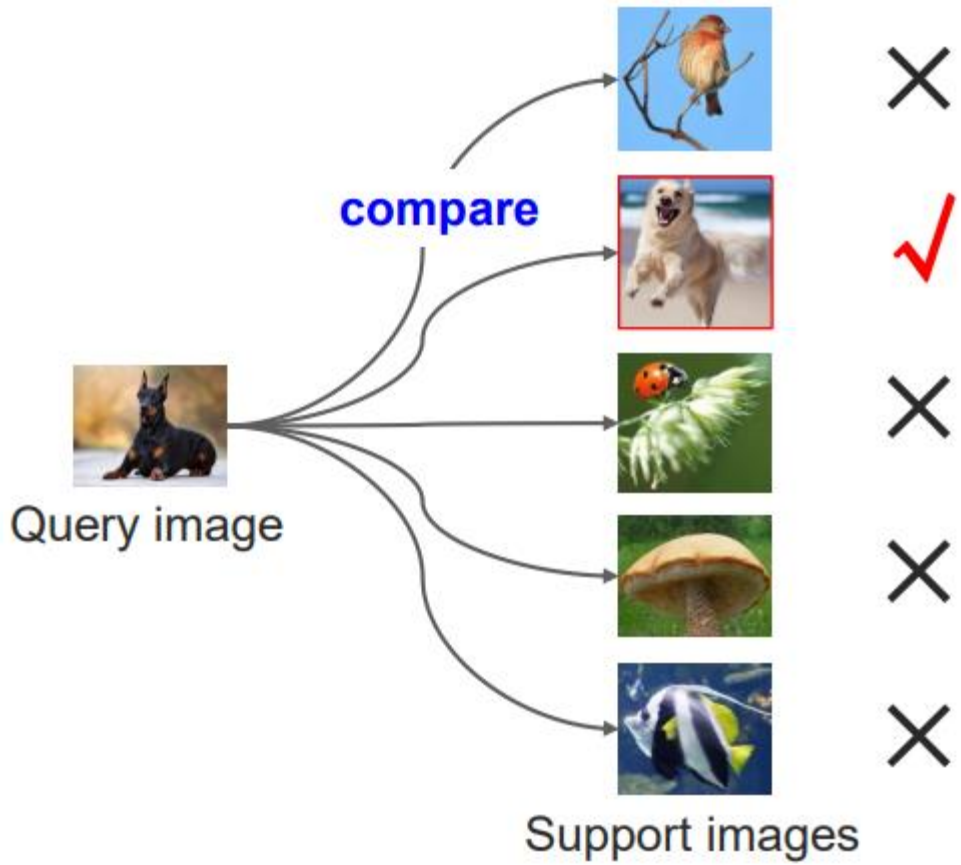




YOO, T. K.; CHOI, J. Y.; KIM, H. K. Feasibility study to improve deep learning in oct diagnosis. 2021.

Classificação few-shot

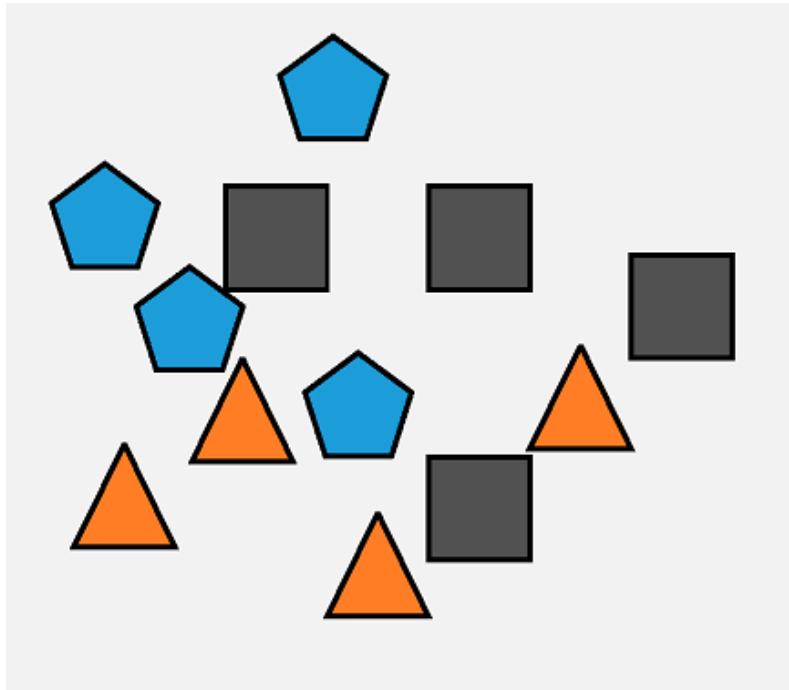
- Há várias técnicas de classificação em FSL;
- Vamos focar na abordagem do metric learning;
- A ideia é classificar imagens por análise de similaridade;

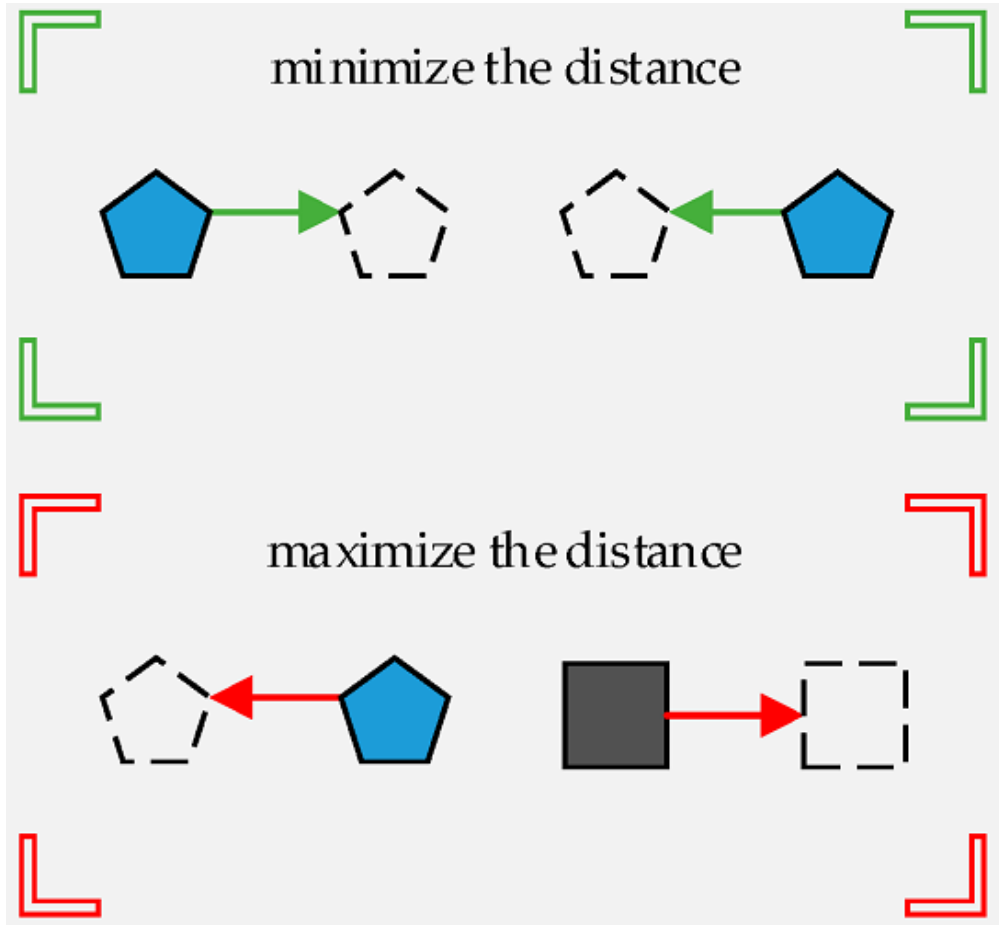


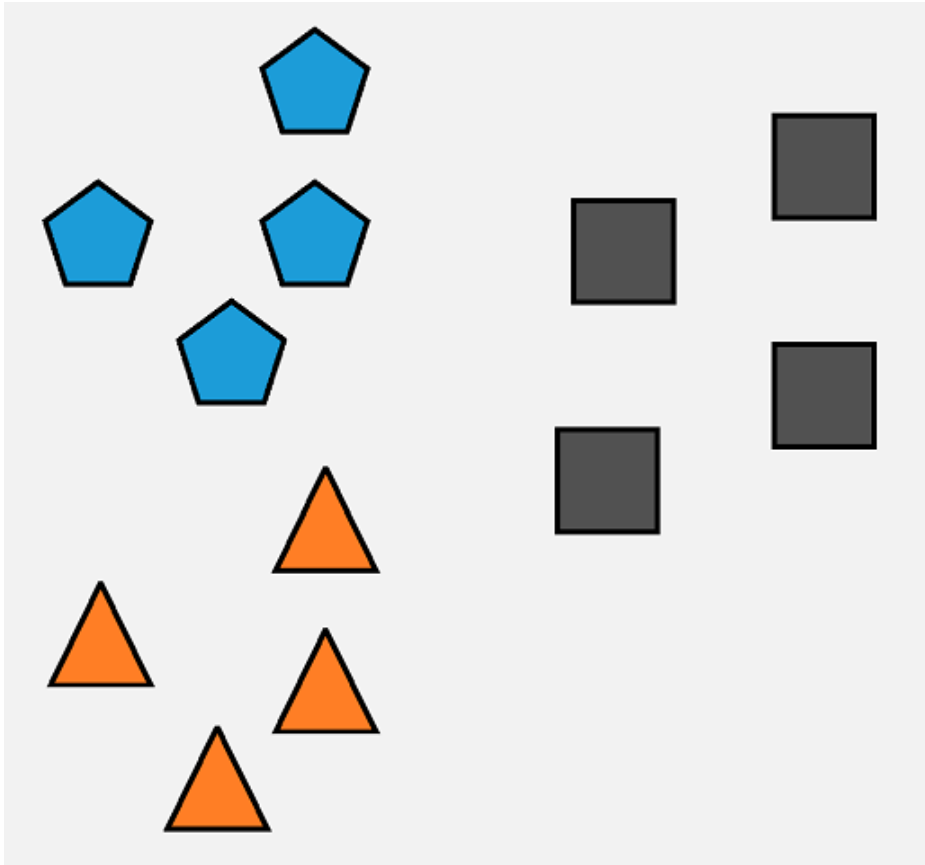
Learn to compare query images with support images

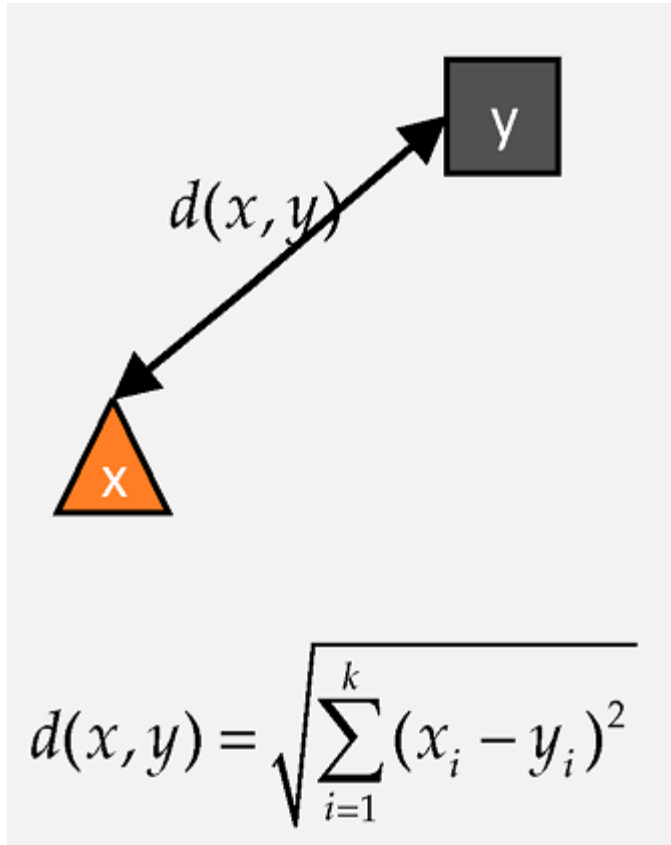
Classificação few-shot

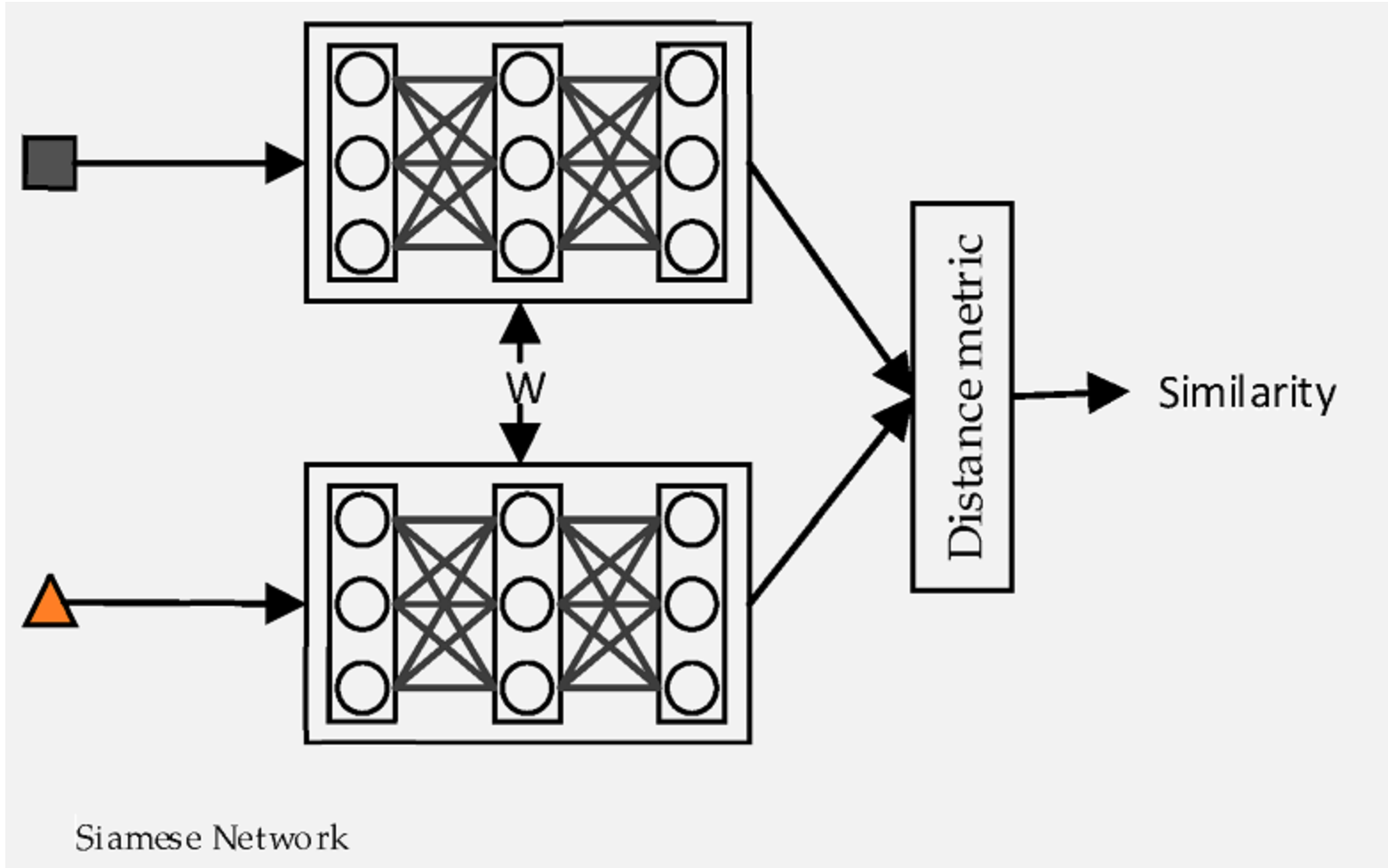
- Para isso, transformamos as imagens em vetores (embeddings);
- E, supomos que os vetores no espaço:
  - Ficam próximos, quando as imagens são similares;
  - Ficam distantes, quando as imagens são dissimilares.
- A classificação é dada por uma métrica (distância).




minimize the distance
maximize the distance




y
d(x, y)
x
$d(x,y) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$



Siamese Network

O que é o embedding?

- É um vetor de características de baixa dimensão;
- Obtido por uma função “image2vector” (encoder):

Imagem → Extração de Características + Redução de Dimensionalidade

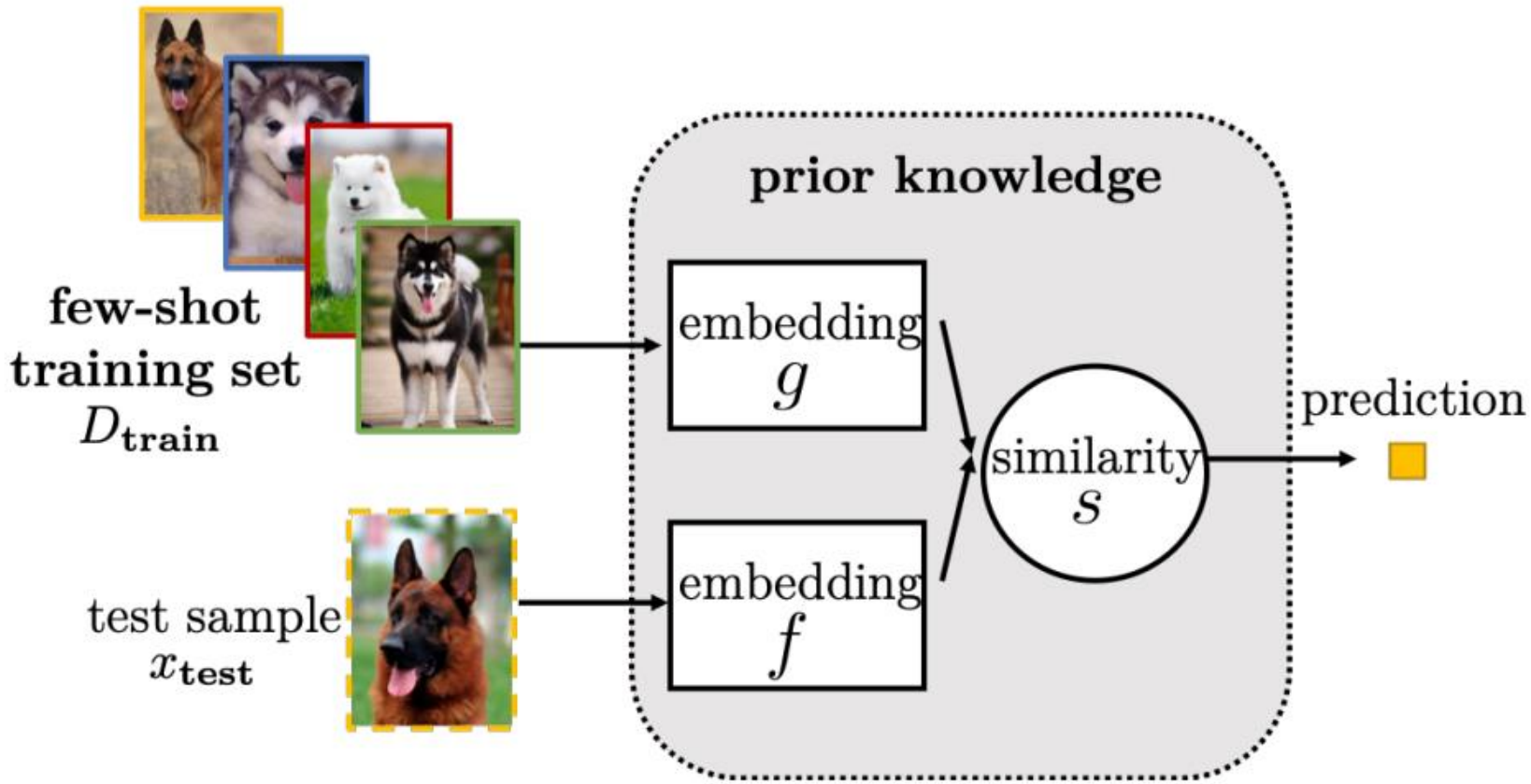
Embedding

Função de embedding

- Alguns modelos usam duas funções de embedding:
  - f_θ para amostras de treinamento;
  - g_ϕ para amostras de teste.
- Outros, usam uma só: $f = g$;

Função de embedding

- Vamos focar em um único encoder f_θ;
- E, considerar um encoder denominado task-invariant;
- Daí, basta encontrarmos um bom conjunto de parâmetros θ;


few-shot
training set
D_{train}
test sample
x_{test}
prior knowledge
embedding
g
embedding
f
similarity
s
prediction

Mas, como é o encoder?

- Há várias arquiteturas para o encoding das imagens;
- A ideia é utilizar modelos bem estabelecidos;
- E, obter a saída do bloco de extração de características;

Mas, como é o encoder?

- Possíveis arquiteturas para o encoder:
  - Inception;
  - Deep Residual Learning (ResNet);
  - 4Conv;
  - Network Architecture Search (NASNet).

Treinamento do modelo

- Splitting das imagens: $D = \{D_{train}, D_{valid}, D_{test}\}$;
- O fluxo de trabalho segue como o usual;
- O modelo de FSL é:
  - Ajustado com D_{train};
  - A cada época, avaliado com D_{valid};
  - Ao final, avaliado com D_{test}.

Treinamento do modelo

- Como f_θ é task-invariant, precisamos de um cuidado especial;
- O modelo precisa habituar-se à generalização;
- Uma solução foi o surgimento dos episódios.

Episódios

- As épocas passam a ser formadas por vários episódios;
- Cada episódio é uma tarefa de classificação diferente;
- A ideia é simular o ambiente de teste durante o treino;

Episódios

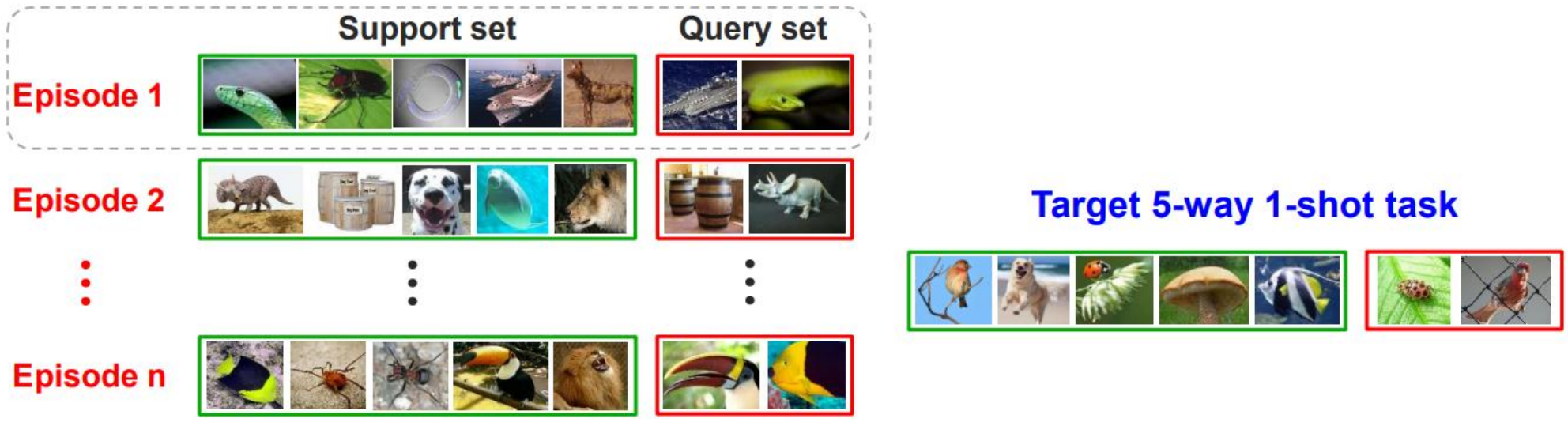
- Um episódio é formado por duas fases:
  - Uma para habituar o modelo com a nova tarefa;
  - Uma para testar a performance na nova tarefa.

Episódios

- As fases são realizadas com os respectivos conjuntos:
  - S: conjunto de suporte → Treino;
  - Q: conjunto de consulta → Teste;
- Os conjuntos devem ser disjuntos;
- Os conjuntos são populados de modo randômico;

Episódios

- Para cada episódio, N_c classes são selecionadas;
- Para S, são selecionados N_s exemplos de cada classe;
- Para Q, são selecionados N_q exemplos de cada classe.


Support set
Query set
Episode 1
Episode 2
Episode n
Target 5-way 1-shot task

Treinamento do modelo

- Os elementos de Q são classificados com base nos:
  - Parâmetros θ do modelo;
  - Elementos do conjunto S.
- O treinamento procede com a otimização da função de likelihood.

Escolha do dataset

- Os conjuntos de dados precisam ter:
  - Uma diversidade boa de classes;
  - Vários exemplos de cada classe.
- Macete: “Data Augmentation” para gerar novas classes.

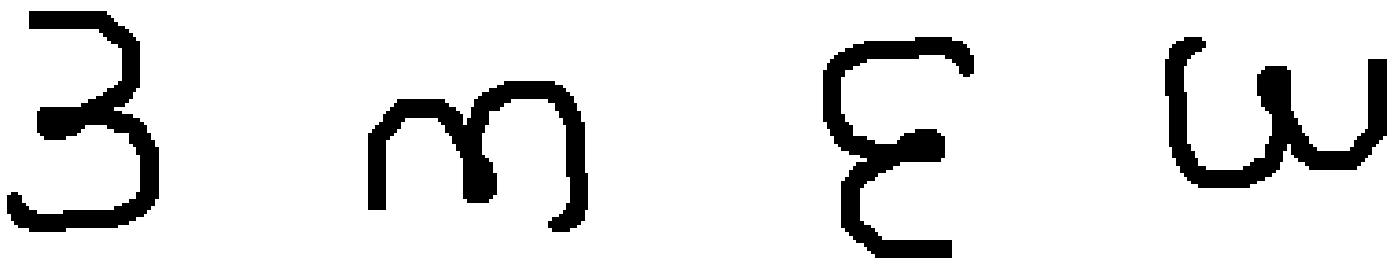
Escolha do dataset

- Datasets bons para FSL:
  - Omniglot;
  - miniImageNet;
  - CUB 200.

Omniglot

- 50 diferentes alfabetos;
- Aprox. 20 caracteres em cada alfabeto;
- Um total de 1.623 classes;
- Com rotações de 90°: 6.492 classes.

rotated Cyrilic character – Ominiglot dataset



miniImageNet

- Derivado do popular ImageNet;
- Criado para tarefas de FSC;
- Um total de 100 classes:
  - 64 para treino;
  - 16 para validação;
  - 20 para teste.
- 600 exemplos por classe.

Redes Prototípicas (RPs)

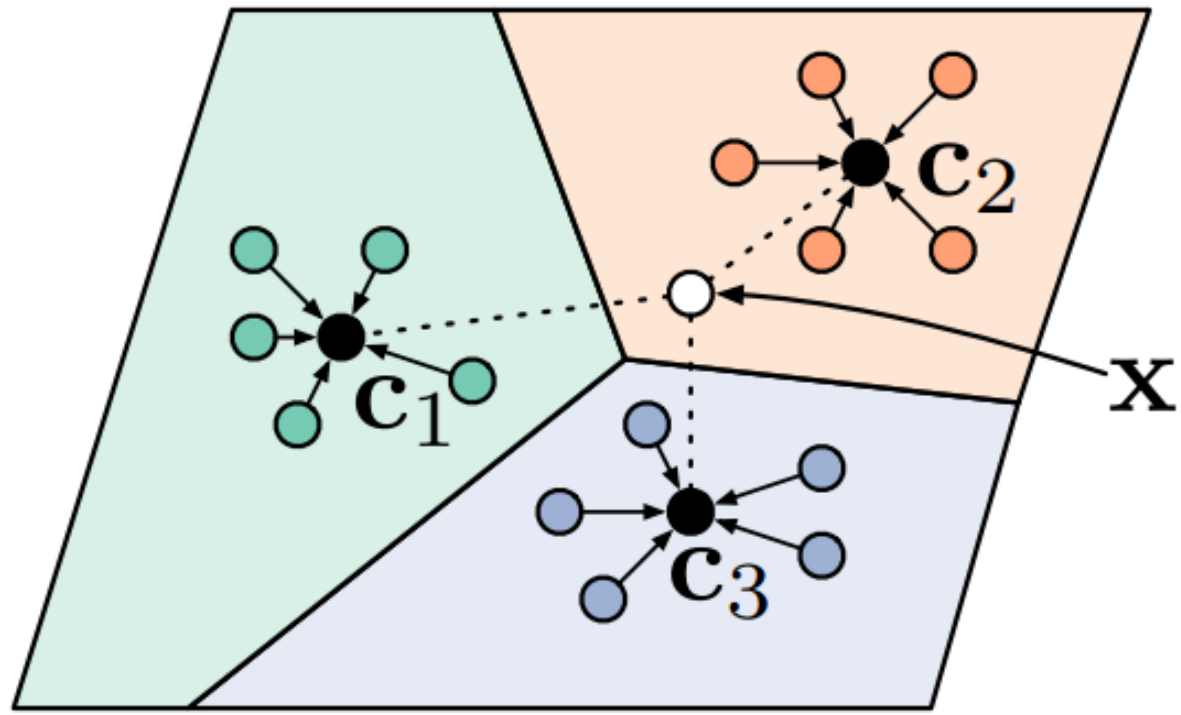
SNELL, J.; SWERSKY, K.; ZEMEL, R. S. Prototypical Networks for FSL. 2017

Ideia das RPs: 1ª etapa

- Para cada uma das A classes, selecionar B imagens;
- Para cada imagem b, extrair um embedding;
- Supor que os vetores no espaço:
  - Ficam próximos, quando as imagens são similares;
  - Ficam distantes, quando as imagens são dissimilares.
- Para cada classe, obter um único representante.

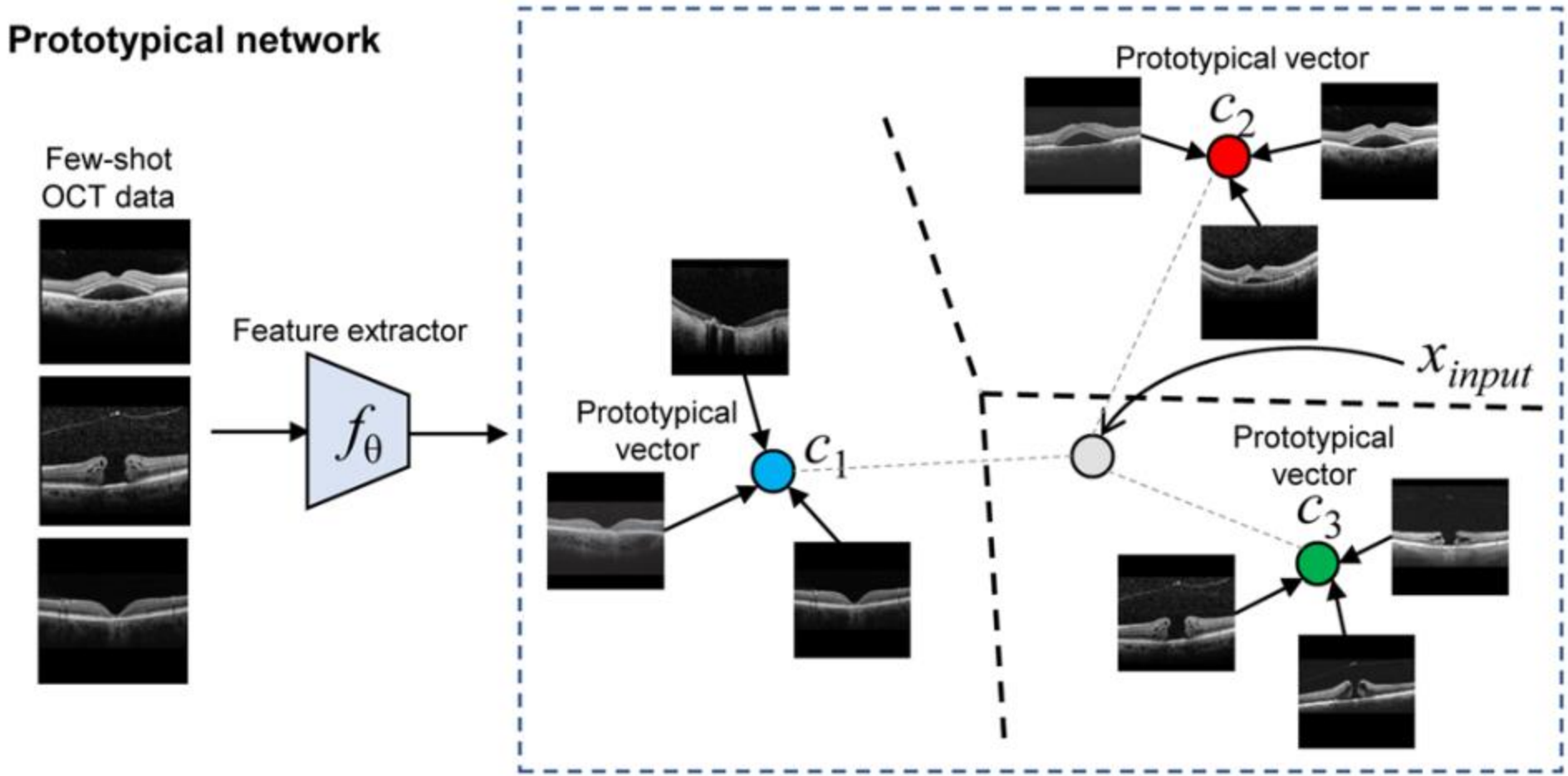
Ideia das RPs: 2ª etapa

- Dada uma imagem x, extrair o seu embedding;
- Medir a distância do vetor ao representante de cada classe;
- Atribuir a x o rótulo do “cluster” mais próximo.


$\mathbf{c}_1$
$\mathbf{c}_2$
$\mathbf{c}_3$
$\mathbf{x}$

Processo de classificação

- As distâncias passam por um softmax:
  - E são convertidas em probabilidades;
  - Com a maior probabilidade indicando a classe.
- A métrica utilizada é distância euclidiana quadrada.


Prototypical network
Few-shot
OCT data
Feature extractor
f_θ
Prototypical vector
c_2
x_{input}
Prototypical
vector
c_1
Prototypical
vector
c_3

Algorithm 1 Training episode loss computation for prototypical networks. N is the number of examples in the training set, K is the number of classes in the training set, $N_C \leq K$ is the number of classes per episode, N_S is the number of support examples per class, N_Q is the number of query examples per class. RANDOMSAMPLE(S, N) denotes a set of N elements chosen uniformly at random from set S.

Input: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each $y_i \in \{1, \ldots, K\}$. $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing all elements $(\mathbf{x}_i, y_i)$ such that $y_i = k$.

Output: The loss J for a randomly generated training episode.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode

for k in $\{1, \ldots, N_C\}$ **do**

  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples

  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples

  $\mathbf{c}_k \leftarrow \frac{1}{N_S} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\phi}(\mathbf{x}_i)$ ▷ Compute prototype from support examples

end for

$J \leftarrow 0$ ▷ Initialize loss

for k in $\{1, \ldots, N_C\}$ **do**

  for $(\mathbf{x}, y)$ in Q_k **do**

    $$J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) \right]$$ ▷ Update loss

  end for

end for

Treinamento das RPs

- A loss é dada pela Negative Log-Likelihood (NLL);
- Expressão para a NLL: $\mathrm{J}(\theta) = -\log P_\theta(y = k|x^*)$;
- A atualização do conjunto θ é feita via SGD com Adam;
- O treino ocorre até que:
  - A loss da valid. pare de melhorar
  - Um número máx. de épocas é atingido.

Arquitetura do encoder

- A arquitetura escolhida foi a 4Conv;
- Composta por quatro blocos convolucionais;
- Cada bloco com uma camada de:
  - Convolução com 64 filtros 3x3;
  - Normalização em lote;
  - ReLU (não linearidade);
  - Max pooling 2x2.

```
def conv_block(in_channels, out_channels):
    return nn.Sequential(
        nn.Conv2d(in_channels, out_channels, 3, padding=1),
        nn.BatchNorm2d(out_channels),
        nn.ReLU(),
        nn.MaxPool2d(2)
    )

encoder = nn.Sequential(
    conv_block(x_dim[0], hid_dim),
    conv_block(hid_dim, hid_dim),
    conv_block(hid_dim, hid_dim),
    conv_block(hid_dim, z_dim),
    Flatten()
)
```

Performance no miniImageNet

- Escolha original: 4Conv;
- Melhores resultados: outras arquiteturas, como ResNet-12.

Network	**Backbone**	**5-way**	
		1-shot	**5-shot**
PN	4Conv	49.42 ± 0.78%	68.20 ± 0.66%
PN	ResNet-12	60.37 ± 0.83%	78.02 ± 0.57%

Vantagens

- Abordagem mais simples e até mais eficiente que outras;
- Atinge resultados do estado da arte;
- Os protótipos reduzem um pouco o efeito de variações intraclasse.

Desvantagens

- São menos adaptáveis que abordagens de meta-learning;
- Consomem bastante tempo devido aos cálculos;
- Algumas imagens são vistas mais que outras no treino;
- Supõe classes balanceadas;
- As RPs não lidam com as variações intraclasse.

Variantes das RPs

Weighted Prototypical Networks (WPN)

- A ideia é utilizar pesos no cálculo do protótipo;
- A cada exemplo de S, é atribuído um peso;
- O objetivo é capturar melhor a representatividade das amostras;

JI, Z.; CHAI, X.; YU, Y.; PANG, Y.; ZHANG, Z. Improved prototypical networks for FSL. 2020.

Weighted Prototypical Networks (WPN)

- A rede é composta por:
  - Uma função de embedding f_θ;
  - Uma função de distribuição de pesos g_φ;
  - Uma função de distância d.

Weighted Prototypical Networks (WPN)

- Para cada classe k:
  - Os embeddings $f_\theta(x)$, com $x \in S_k$, são obtidos;
  - Os pesos $\mathrm{w} = g_\varphi\big(f_\theta(x)\big)$ são determinados;
  - Um protótipo é calculado por $C_k = \sum_1^{N_s} w_n f_\theta(x_n)$.
- A classificação segue pelo cálculo das distâncias.

Task-Adaptive Projection Networks (TapNet)

- A ideia é criar um espaço de classificação “task-dependent”;
- Um espaço novo é construído para cada tarefa;
- O objetivo é melhorar a generalização;

YOON, S. W.; SEO, J.; MOON, J. TapNet for FSL. 2019.

Task-Adaptive Projection Networks (TapNet)

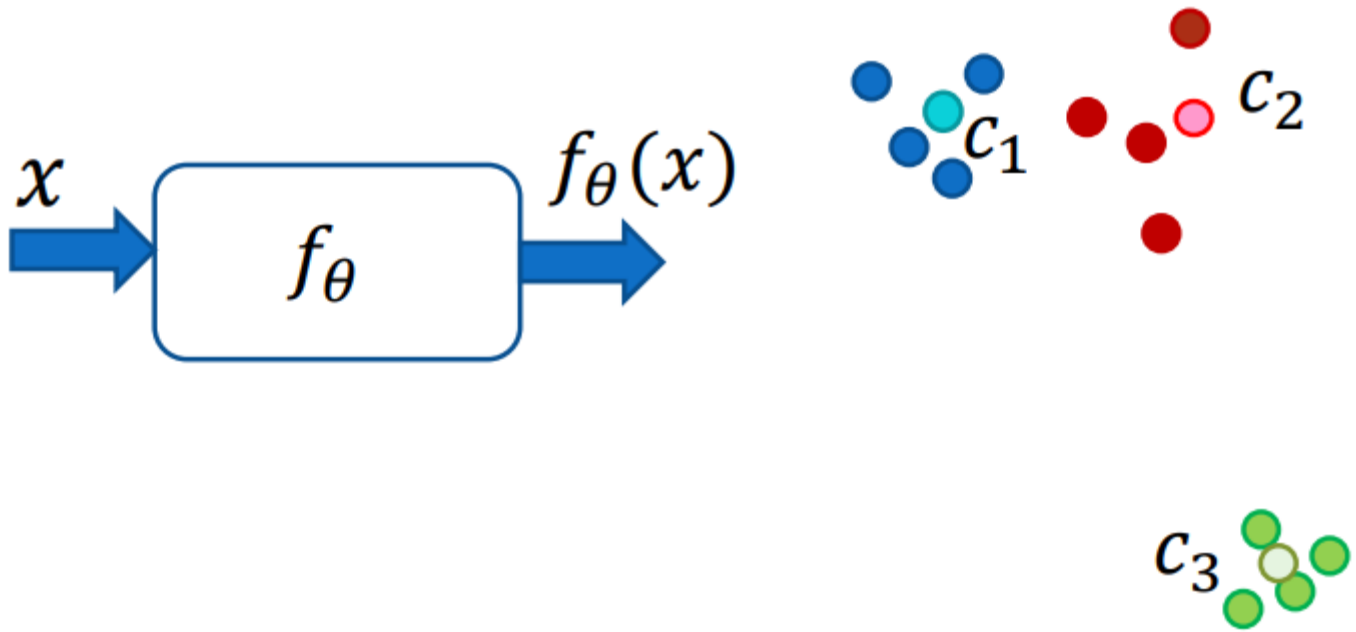
- Isso é atingido por meio de:
  - Uma função de embedding f_θ;
  - Um conjunto de vetores de referência Φ;
  - Uma projeção ou mapeamento M;
  - Uma função de distância d.

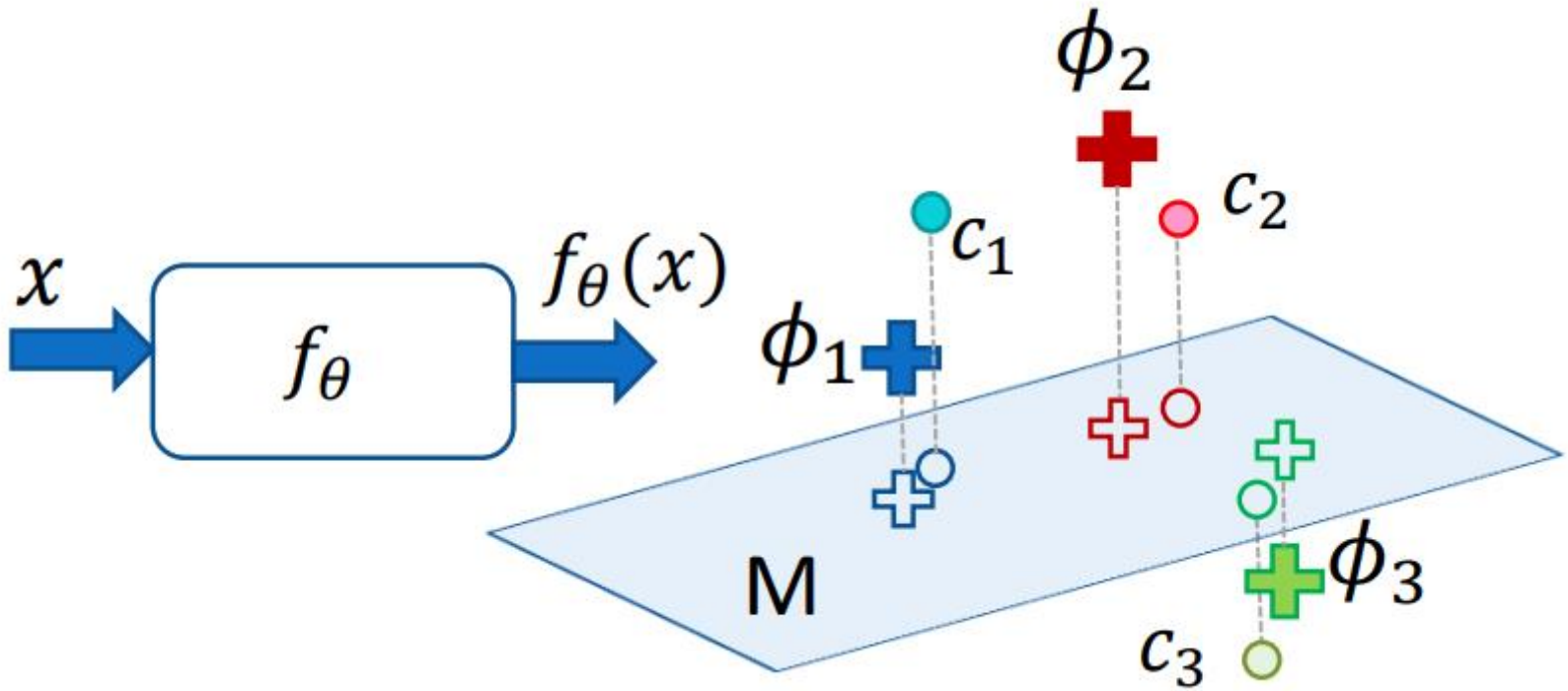
Task-Adaptive Projection Networks (TapNet)

- Para cada classe k:
  - Os embeddings $f_\theta(x)$, com $x \in S_k$, são obtidos;
  - O protótipo c_k é calculado;
  - Um vetor de referência ϕ_k é atribuído.

Task-Adaptive Projection Networks (TapNet)

- Com M, obtemos um espaço em que:
  - O protótipo c_k e a referência ϕ_k fiquem bem alinhados (próximos);
  - O protótipo c_k e a referência ϕ_l, com $l \neq k$, fiquem bem separados.
- A classificação é dada por: $d(M(f_\theta(x^*)), M(\phi_k))$.


x
f_θ
$f_\theta(x)$
c_1
c_2
c_3


x
f_θ
$f_\theta(x)$
ϕ_1
ϕ_2
ϕ_3
c_1
c_2
c_3
M

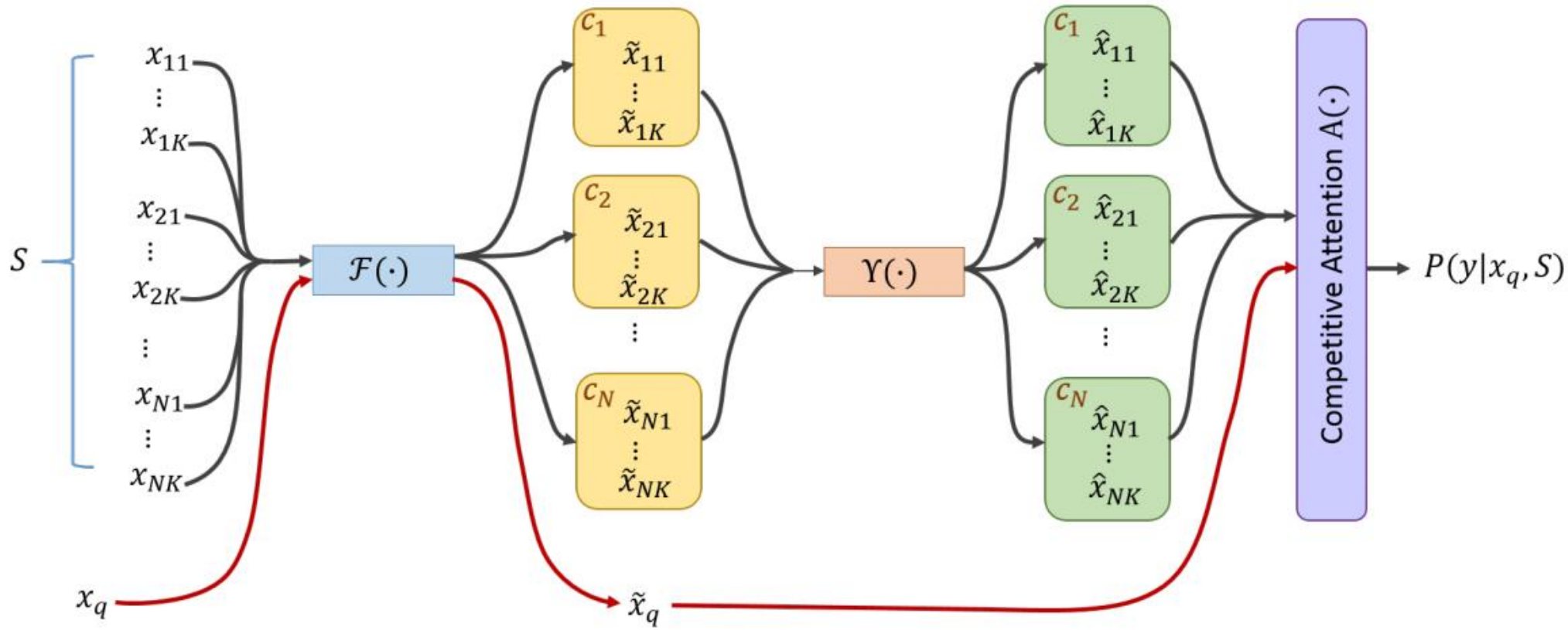
Class Support Networks (CS-Nets)

- A ideia é aumentar a representatividade:
  - Dos embeddings de suporte;
  - Dos protótipos utilizados para classificar.
- Uma distribuição melhor do suporte é formada;
- Protótipos são escolhidos de acordo com o ponto de query;

LIU, J.; GIBSON, S. J.; OSADCHY, M. Learning to Support. 2018.

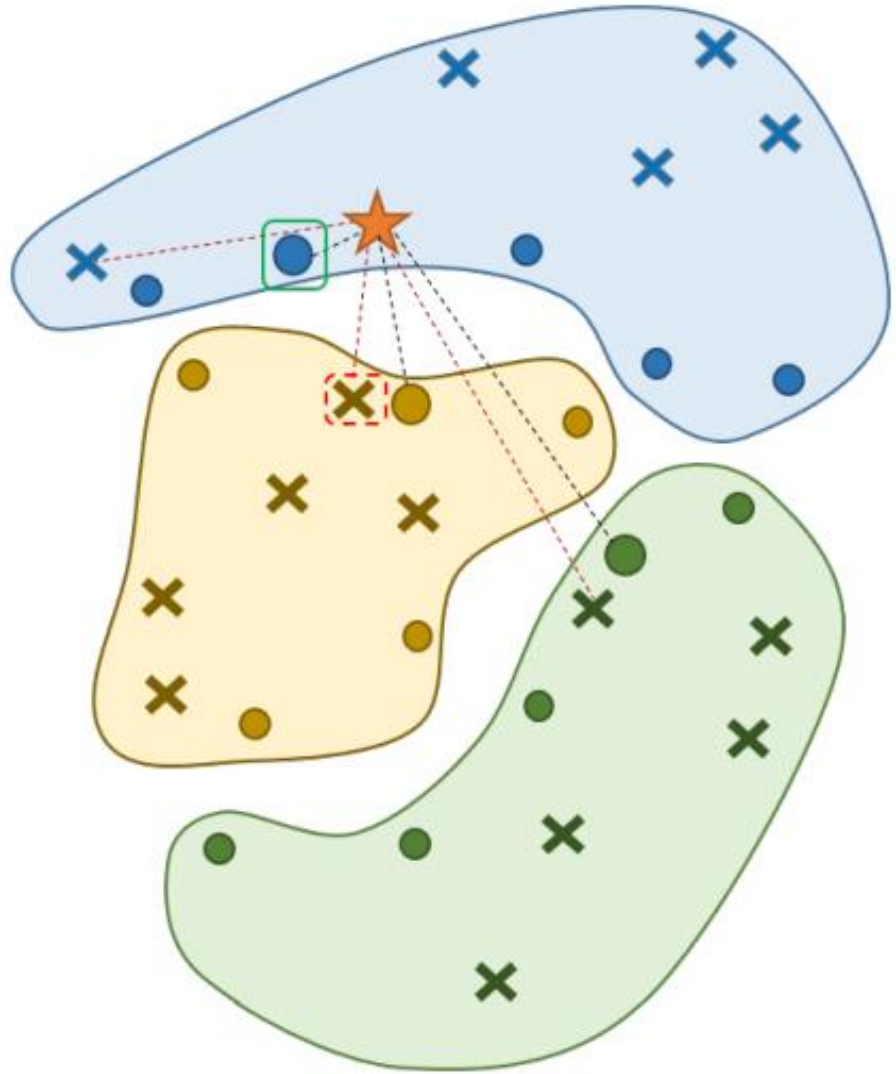
Class Support Networks (CS-Nets)

- A rede é composta por:
  - Uma função de embedding f_θ;
  - Uma função de class support embedding γ_ϕ;
  - Um mecanismo de atenção A;
  - Uma função de distância d.


S
x_{11}
x_{1K}
x_{21}
x_{2K}
x_{N1}
x_{NK}
x_q
$\mathcal{F}(\cdot)$
c_1 $\tilde{x}_{11}$ $\tilde{x}_{1K}$
c_2 $\tilde{x}_{21}$ $\tilde{x}_{2K}$
c_N $\tilde{x}_{N1}$ $\tilde{x}_{NK}$
$\tilde{x}_q$
$\Upsilon(\cdot)$
c_1 $\hat{x}_{11}$ $\hat{x}_{1K}$
c_2 $\hat{x}_{21}$ $\hat{x}_{2K}$
c_N $\hat{x}_{N1}$ $\hat{x}_{NK}$
Competitive Attention $A(\cdot)$
$P(y|x_q, S)$

Class Support Networks (CS-Nets)

- Classificação de x^* por meio de A:
  - Os pontos $\gamma_\phi(f_\theta(x_k))$ competem para ser o protótipo da classe k;
  - Em cada classe, o ponto mais próximo de x^* ganha a competição;
  - x^* é classificado de acordo com o protótipo mais próximo.



Performance no miniImageNet

Network	Backbone	5-way	
		1-shot	5-shot
PN	4Conv	49.42 ± 0.78%	68.20 ± 0.66%
PN	ResNet-10	51.98 ± 0.84%	72.64 ± 0.64%
PN	ResNet-12	60.37 ± 0.83%	**78.02 ± 0.57%**
WPN	ResNet-10	-	74.20%
TapNet	ResNet-12	**61.65 ± 0.15%**	76.36 ± 0.10%
CS-Net	4Conv	52.56 ± 0.45%	69.05 ± 0.36%

Outras variações interessantes

- Rede Prototípica Gaussiana / Stochastic Prototype Embeddings;
- Rede Prototípica Multimodal;
- Rede Prototípica Semi-supervisionada.

Referências

Referências

CHEN, W.-Y.; LIU, Y.-C.; KIRA, Z.; WANG, Y.-C. F.; HUANG, J.-B. A Closer Look at Few-shot Classification. 2020. Disponível em: https://arxiv.org/pdf/1904.04232.pdf.

FORT, S. Gaussian Prototypical Networks for Few-Shot Learning on Omniglot. 2017. Disponível em: https://arxiv.org/pdf/1708.02735.pdf.

GOOGLE CLOUD. Overview: Extracting and serving feature embeddings for machinelearning. 2020. Disponível em: https://cloud.google.com/solutions/machine-learning/overview-extracting-and-serving-feature-embeddings-for-machine-learning#extracting_embeddings_from_documents_or_images.

JI, Z.; CHAI, X.; YU, Y.; PANG, Y.; ZHANG, Z. Improved prototypical networks for few-shot learning. Pattern Recognition Letters, v. 140, p. 81–87, 2020. ISSN 0167-8655. Disponível em:https://www.sciencedirect.com/science/article/pii/S016786552030261O.

KARPATHY, A. t-SNE visualization of CNN codes. Disponível em: https://cs.stanford.edu/people/karpathy/cnnembed/.

Referências

KAYA, M.; BİLGE, H. Deep metric learning: A survey. Symmetry, v. 11, n. 9, 2019. ISSN2073-8994. Disponível em: https://www.mdpi.com/2073-8994/11/9/1066.

LAENEN, S.; BERTINETTO, L. On Episodes, Prototypical Networks, and Few-shot Learning. 2020. Disponível em: https://arxiv.org/pdf/2012.09831.pdf.

LAKE, B. M.; SALAKHUTDINOV, R.; TENENBAUM, J. B. Human-level concept learning through probabilistic program induction. Science, American Association for the Advancement of Science, v. 350, n. 6266, p. 1332–1338, 2015. ISSN 0036-8075. Disponível em: https://science.sciencemag.org/content/350/6266/1332.

LAROCHELLE, H. Few-shot Learning with Meta-Learning: Progress made and challenges ahead. Disponível em: https://www.youtube.com/watch?v=lz0ekIVfoFs.

LIU, J.; GIBSON, S. J.; OSADCHY, M. Learning to Support: Exploiting Structure Information in Support Sets for One-Shot Learning. 2018. Disponível em: https://arxiv.org/pdf/1808.07270.pdf.

Referências

LUO, J. Advances in few-shot learning. Disponível em: https://www.eecs.ucf.edu/~gqi/publications/IJCAI19--few-shot_learning.pdf.

NIELLY, C. Few-shot Learning with Prototypical Networks: Learn to code a few-shot learning algorithm on the omniglot dataset. 2020. Disponível em: https://towardsdatascience.com/few-shot-learning-with-prototypical-networks-87949de03ccd.

OSADCHY, R. Learning from Small Data. Disponível em: https://erez.weizmann.ac.il/pls/htmldb/f?p=101/58///NO/RP/P58_CODE_/_P58_FILE/9305_/_Y.

PAHDE, F.; PUSCAS, M.; KLEIN, T.; NABI, M. Multimodal Prototypical Networks for Few-shot Learning. 2020. Disponível em: https://arxiv.org/pdf/2011.08899.pdf.

REN, M.; TRIANTAFILLOU, E.; RAVI, S.; SNELL, J.; SWERSKY, K.; TENENBAUM, J. B.;LAROCHELLE, H.; ZEMEL, R. S. Meta-Learning for Semi-Supervised Few-Shot Classification. 2018. Disponível em: https://arxiv.org/pdf/1803.00676.pdf.

Referências

SCOTT, T. R.; RIDGEWAY, K.; MOZER, M. C. Stochastic Prototype Embeddings. 2019. Disponível em: https://arxiv.org/pdf/1909.11702.pdf.

SNELL, J. Frontiers of Metric-based Few-shot Learning. Disponível em: https://www.youtube.com/watch?v=IxJ9dTHhvSg.

SNELL, J.; SWERSKY, K.; ZEMEL, R. S. Prototypical Networks for Few-shot Learning. 2017. Disponível em: https://arxiv.org/pdf/1703.05175.pdf.

VINYALS, O.; BLUNDELL, C.; LILLICRAP, T.; KAVUKCUOGLU, K.; WIERSTRA, D. Matching Networks for One Shot Learning. 2017. Disponível em: https://arxiv.org/pdf/1606.04080.pdf.

WANG, Y.; YAO, Q.; KWOK, J.; NI, L. M. Generalizing from a Few Examples: A Survey on Few-Shot Learning. 2020. Disponível em: https://arxiv.org/pdf/1904.05046.pdf.

Referências

YAVARI, N. Few-Shot Learning with Deep Neural Networks for Visual Quality Control: Evaluations on a Production Line. 55 p. Dissertação (Mestrado) — KTH, School of Electrical Engineering and Computer Science (EECS), 2020. Disponível em: http://kth.diva-portal.org/smash/get/diva2:1473064/FULLTEXT01.pdf.

YOO, T. K. Data for: Improved accuracy in oct diagnosis of rare retinal disease using few-shot learning with generative adversarial networks. Mendeley Data, v. 2, 2020. Disponível em: https://data.mendeley.com/datasets/btv6yrdbmv/2.

YOO, T. K.; CHOI, J. Y.; KIM, H. K. Feasibility study to improve deep learning in oct diagnosis of rare retinal diseases with few-shot classification. Medical & Biological Engineering & Computing, Springer, v. 59, n. 2, p. 401–415, 2021. Disponível em: https://link.springer.com/content/pdf/10.1007/s11517-021-02321-1.pdf.

YOON, S. W.; SEO, J.; MOON, J. TapNet: Neural Network Augmented with Task-Adaptive Projection for Few-Shot Learning. 2019. Disponível em: https://arxiv.org/pdf/1905.06549.pdf.

Referências

ZHANG, C.; CAI, Y.; LIN, G.; SHEN, C. DeepEMD: Differentiable Earth Mover's Distance for Few-Shot Learning. 2020. Disponível em: https://arxiv.org/pdf/2003.06777v1.pdf.

ZI, W.; PRINCE, S.; GHORAIE, L. S. Tutorial 2: Few-shot learning and meta-learning i. 2019. Disponível em: https://www.borealisai.com/en/blog/tutorial-2-few-shot-learning-and-meta-learning-i/.

Obrigado pela atenção!
Perguntas?