5. Learning to Compare: Relation Network for Few-Shot Learning

논문 : https://arxiv.org/pdf/1711.06025.pdf

요약

- 4. matching network의 episodic training 기법을 채용
- 모델 f 로 query data와 sample(support) data의 feature vector 연산
- query data와 sample data별 feature vector를 concat한뒤 모델 g로 relation score (0~1 사이 값)을 출력
    - 1~4 논문들과 다르게 trainable non-linear metric based meta learning을 제안
- zero-shot / one-shot / few-shot(5-shot)에서 적용할 수 있는 구조를 제안

- 기존의 metric beasd 기법들 (prototypical network, matching network)은 고정된 linear metric distance(euclidean distance, cosine distance)를 사용함
    - prototypical network 논문에서는 'linear metric을 사용하더라도 non linear를 학습할수 있음'을 다른 논문이 주장했기 때문에 괜찮다했음
- 본 논문은 trainable non-linear metric을 사용하는 metric based meta learning을 제안함 (그림 1)
    - embedding module f로 sample(support) data(보라색, 하늘색, 회색, 연분홍색, 연두색)와 query data(노란색)의 feature vector 연산
    - query data의 feature vector와 sample data의 feature vector를 concatenation한뒤 relation module g로 관계점수를 연산
    - 가장 높은 점수의 클래스로 최종 출력
- 본 논문도 episode training 으로 학습을 진행
    - class/query 선택은 random sampling으로 다른 논문과 동일

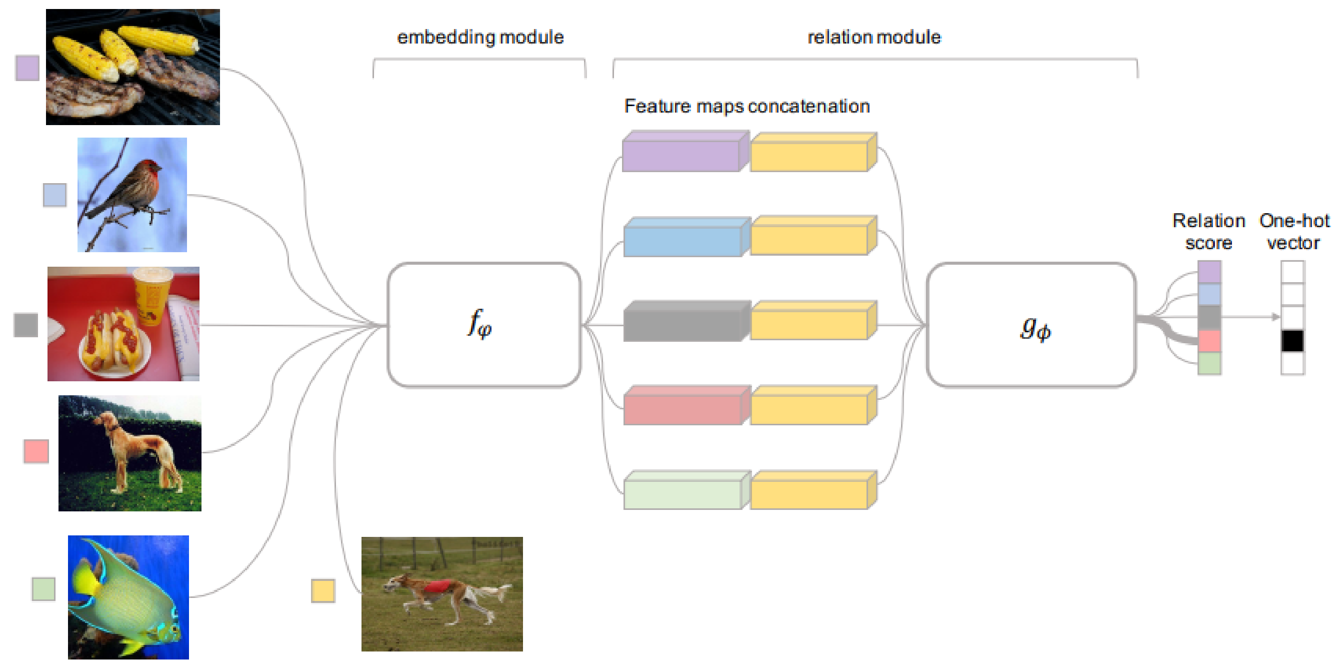

Figure 1: Relation Network architecture for a 5-way 1-shot problem with one query example.

기호	설명
C	에피소드 별 클래스 개수 5일 경우 5-way N-shot learning
$\mathcal{C}$	Concatenation function
f_φ	feature vector를 연산하는 embedding module

g_ϕ	두 feature vector가 연결된 값을 입력으로 relation score를 연산 0에 가까울 수록 관계가 없음 (mismatched pair) 1에 가까울 수록 서로 관계가 있음 (matched pair)
$r_{i,j}$	episode내에서 sample dataset 중 i번째 sample data와 query dataset 중 j번째 query data간의 relation score

one-shot / few-shot learning

- one-shot / few-shot learning에서는 식 1을 사용해서 relation score를 연산
- few-shot learning과 같이 동일 class에 2개이상의 sample data가 있는 경우
    - class 별 feature vector들을 element-wise sum한 vector를 사용
- 모델 구조 (그림 2)
    - embedding module (feature concatenation 아래까지) : 기존 논문들이 제안한 구조를 사용
    - feature concatenation : sample feature와 query feature를 채널로 연결

$$r_{i,j} = g_\phi(\mathcal{C}(f_\varphi(x_i), f_\varphi(x_j))), \quad i = 1, 2, \ldots, C \qquad (1)$$

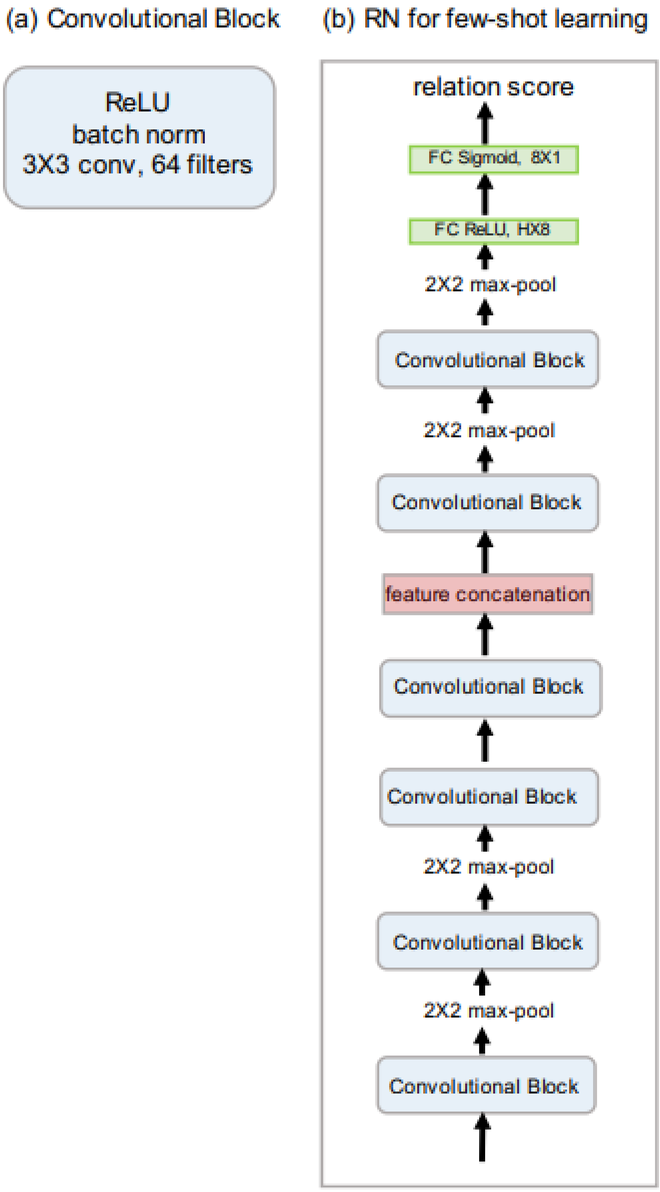

Figure 2: Relation Network architecture for few-shot learning (b) which is composed of elements including convolutional block (a).

zero-shot learning

- zero-shot learning 특성상 query data와 동일한 형태의 sample data를 사용하는 것이 아님
    - ex) query data가 image인 경우 sample data는 image data description 형태
- 다른 형태의 data에 대한 feature vector를 하나의 모델이 연산하기엔 힘들기 때문에 동일 구조에 embedding module을 두개 운용하는 방식을 제안 (식 3)
- 모델 구조 (그림 3)
    - sample data의 메타정보는 간단한 Fully Connected Layer로 연산
    - query data는 DNN (imagenet으로 pre-train된 Inception/ResNet)으로 연산

$$r_{i,j} = g_\phi(\mathcal{C}(f_{\varphi_1}(v_c), f_{\varphi_2}(x_j))), \quad i = 1, 2, \ldots, C \qquad (3)$$

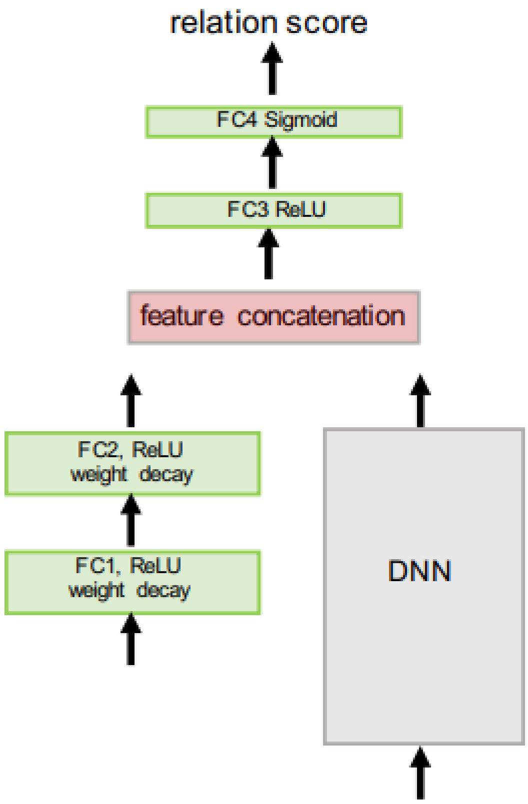

Figure 3: Relation Network architecture for zero-shot learning.

Objective function

- Mean Square Error (MSE)를 사용

The choice of MSE is somewhat non-standard. Our problem may seem to be a classification problem with a label space $\{0, 1\}$. However conceptually we are predicting relation scores, which can be considered a regression problem despite that for ground-truth we can only automatically generate $\{0, 1\}$ targets.

$$\varphi, \phi \leftarrow \underset{\varphi, \phi}{\operatorname{argmin}} \sum_{i=1}^{m} \sum_{j=1}^{n} (r_{i,j} - \mathbf{1}(y_i == y_j))^2 \quad (2)$$

실험

- 실험 종류
  - few-shot learning 성능 비교
    - 사용 데이터 셋
      - Omniglot (표 1)
        - 사용한 augmentation 기법
          - rotation (90,180,270도 회전)
        - 학습 시 M-way N-shot 마다 query image개수를 달리함
          - 한번의 episode/minibatch를 **100**으로 설정하기 위함
            - query image num * M-way + N-shot * N-way = 100
          - 5-way 1-shot : 19 query image
          - 5-way 5-shot : 15 query image
          - 20-way 1-shot : 10 query image
          - 20-way 5-shot : 5 query image
        - 테스트는 1000개의 episode로 진행
          - episode는 역시 random sampling
      - miniImagenet (표 2)
        - 학습 시 M-way N-shot 마다 episode/minibatch 당 80개의 이미지를 사용
          - 5-way 1-shot : 15 query image
          - 5-way 5-shot : 10 query image
        - 84 x 84 사이즈로 resizing
        - 테스트는 평균 600개 이상의 episode로 진행
          - episode는 역시 random sampling

Model	**Fine Tune**	**5-way Acc.**		**20-way Acc.**	
		1-shot	5-shot	1-shot	5-shot
MANN [32]	N	82.8%	94.9%	-	-
CONVOLUTIONAL SIAMESE NETS [20]	N	96.7%	98.4%	88.0%	96.5%
CONVOLUTIONAL SIAMESE NETS [20]	Y	97.3%	98.4%	88.1%	97.0%
MATCHING NETS [39]	N	98.1%	98.9%	93.8%	98.5%
MATCHING NETS [39]	Y	97.9%	98.7%	93.5%	98.7%
SIAMESE NETS WITH MEMORY [18]	N	98.4%	99.6%	95.0%	98.6%
NEURAL STATISTICIAN [8]	N	98.1%	99.5%	93.2%	98.1%
META NETS [27]	N	99.0%	-	97.0%	-
PROTOTYPICAL NETS [36]	N	98.8%	99.7%	96.0%	98.9%
MAML [10]	Y	98.7 ± 0.4%	**99.9 ± 0.1%**	95.8 ± 0.3%	98.9 ± 0.2%
RELATION NET	N	**99.6 ± 0.2%**	**99.8± 0.1%**	**97.6 ± 0.2%**	**99.1± 0.1%**

Table 1: Omniglot few-shot classification. Results are accuracies averaged over 1000 test episodes and with 95% confidence intervals where reported. The best-performing method is highlighted, along with others whose confidence intervals overlap. '-': not reported.

Model	FT	5-way Acc.	
		1-shot	5-shot
MATCHING NETS [39]	N	$43.56 \pm 0.84\%$	$55.31 \pm 0.73\%$
META NETS [27]	N	$49.21 \pm 0.96\%$	-
META-LEARN LSTM [29]	N	$43.44 \pm 0.77\%$	$60.60 \pm 0.71\%$
MAML [10]	Y	$48.70 \pm 1.84\%$	$63.11 \pm 0.92\%$
PROTOTYPICAL NETS [36]	N	$49.42 \pm 0.78\%$	$\mathbf{68.20 \pm 0.66\%}$
RELATION NET	N	$\mathbf{50.44 \pm 0.82\%}$	$65.32 \pm 0.70\%$

Table 2: Few-shot classification accuracies on *mini*Imagenet. All accuracy results are averaged over 600 test episodes and are reported with 95% confidence intervals, same as [36]. For each task, the best-performing method is highlighted, along with any others whose confidence intervals overlap. '-': not reported.

zero-shot learning 성능 비교

- 실험 방법
  - old 버전 (표 3)
    - 테스트셋으로 unseen data만 존재한다는 가정
  - new 버전 (표 4)
    - 테스트셋으로 unseen/seen data가 혼합되있는 가정
- 사용 데이터 셋
  - Animals with Attributes (AwA) (표 3, 4)
    - 50종류 동물에 30,745 개 이미지
    - 논문이 지정한 attribute vector 크기 : 85
  - Caltech-UCSD Birds-200-2011 (CUB) (표 3, 4)
    - 200종류 새의 11,788 개 이미지
    - 논문이 지정한 attribute vector 크기 : 312

Model	F	SS	AwA 10-way 0-shot	CUB 50-way 0-shot
SJE [3]	F_G	A	66.7	50.1
ESZSL [31]	F_G	A	76.3	47.2
SSE-RELU [46]	F_V	A	76.3	30.4
JLSE [47]	F_V	A	80.5	42.1
SYNC-STRUCT [6]	F_G	A	72.9	54.5
SEC-ML [5]	F_V	A	77.3	43.3
PROTO. NETS [36]	F_G	A	-	54.6
DEVISE [11]	N_G	A/W	56.7/50.4	33.5
SOCHER *et al.* [37]	N_G	A/W	60.8/50.3	39.6
MTMDL [43]	N_G	A/W	63.7/55.3	32.3
BA *et al.* [25]	N_G	A/W	69.3/58.7	34.0
DS-SJE [30]	N_G	A/D	-	50.4/ 56.8
SAE [21]	N_G	A	84.7	61.4
DEM [45]	N_G	A/W	**86.7**/78.8	58.3
RELATION NET	N_G	A	84.5	**62.0**

Table 3: Zero-shot classification accuracy (%) comparison on AwA and CUB (hit@1 accuracy over all samples) under the old and conventional setting. SS: semantic space; A: attribute space; W: semantic word vector space; D: sentence description (only available for CUB). F: how the visual feature space is computed; For non-deep models: F_O if overfeat [34] is used; F_G for GoogLeNet [38]; and F_V for VGG net [35]. For neural network based methods, all use Inception-V2 (GoogLeNet with batch normalisation) [38, 17] as the DNN image imbedding subnet, indicated as N_G.

	AwA1				AwA2				CUB			
	ZSL	GZSL			ZSL	GZSL			ZSL	GZSL		
Model	T1	u	s	H	T1	u	s	H	T1	u	s	H
DAP [24]	44.1	0.0	88.7	0.0	46.1	0.0	84.7	0.0	40.0	1.7	67.9	3.3
CONSE [28]	45.6	0.4	88.6	0.8	44.5	0.5	90.6	1.0	34.3	1.6	**72.2**	3.1
SSE [46]	60.1	7.0	80.5	12.9	61.0	8.1	82.5	14.8	43.9	8.5	46.9	14.4
DEVISE [11]	54.2	13.4	68.7	22.4	59.7	17.1	74.7	27.8	52.0	23.8	53.0	32.8
SJE [3]	65.6	11.3	74.6	19.6	61.9	8.0	73.9	14.4	53.9	23.5	59.2	33.6
LATEM [41]	55.1	7.3	71.7	13.3	55.8	11.5	77.3	20.0	49.3	15.2	57.3	24.0
ESZSL [31]	58.2	6.6	75.6	12.1	58.6	5.9	77.8	11.0	53.9	12.6	63.8	21.0
ALE [2]	59.9	16.8	76.1	27.5	62.5	14.0	81.8	23.9	54.9	23.7	62.8	34.4
SYNC [6]	54.0	8.9	87.3	16.2	46.6	10.0	90.5	18.0	55.6	11.5	70.9	19.8
SAE [21]	53.0	1.8	77.1	3.5	54.1	1.1	82.2	2.2	33.3	7.8	57.9	29.2
DEM [45]	**68.4**	**32.8**	84.7	**47.3**	**67.1**	**30.5**	86.4	45.1	51.7	19.6	54.0	13.6
RELATION NET	68.2	31.4	**91.3**	46.7	64.2	30.0	**93.4**	**45.3**	**55.6**	**38.1**	61.1	**47.0**

Table 4: Comparative results under the GBU setting. Under the conventional ZSL setting, the performance is evaluated using per-class average Top-1 (**T1**) accuracy (%), and under GZSL, it is measured using **u** = **T1** on unseen classes, **s** = **T1** on seen classes, and **H** = harmonic mean.

왜 Relation Network가 잘되는가?

- 기존기법들은?
  - fixed metric approaches
    - 학습이 embedding module에서만 이뤄짐
    - 유클리드 거리, 코사인 유사도 같은 고정된 거리계산함수만 사용함
  - metric learning approaches
    - shllow(linear) mahalanobis metric을 학습하는데 주력
    - feature vector연산은 고정해놓음
- 제안 기법은?
  - 둘다 학습하게 해서 상호 보완적이다~
  - 시각화 결과 (그림 4, 5)
    - 그림 4
      - (a) 2D 데이터 모음 (노란색이 유사함을 표현한 1, 그외 0)
      - (b) 2D 데이터를 입력으로 feature vector 연산 + realation network를 통과했을 때 결과
      - (c) 2D 데이터를 입력으로 mahalanobis metric을 통과했을 때 결과
      - (d) 2D 데이터를 입력으로 feature vector 연산(Fully Connected Layer 2개) + mahalanobis metric을 통과했을 때 결과
    - 그림 5
      - 1열) feature vector 크기를 2차원으로 가정했을 때 omniglot dataset의 feature vector 분포 표현
        - 노란색 : query data
        - 청록색 : matched data
        - 보라색 : mistached data
        - 일반적인 거리 계산 함수(유클리드 거리, 코사인 유사도, 등)을 적용하면 에러가 날것
      - 2열) relation module의 결과값을 2D PCA로 표현
        - 노란색 : matched data
        - 보라색 : mismatched data
        - 논문이 제안한 relation module이 잘 갈라주드라~

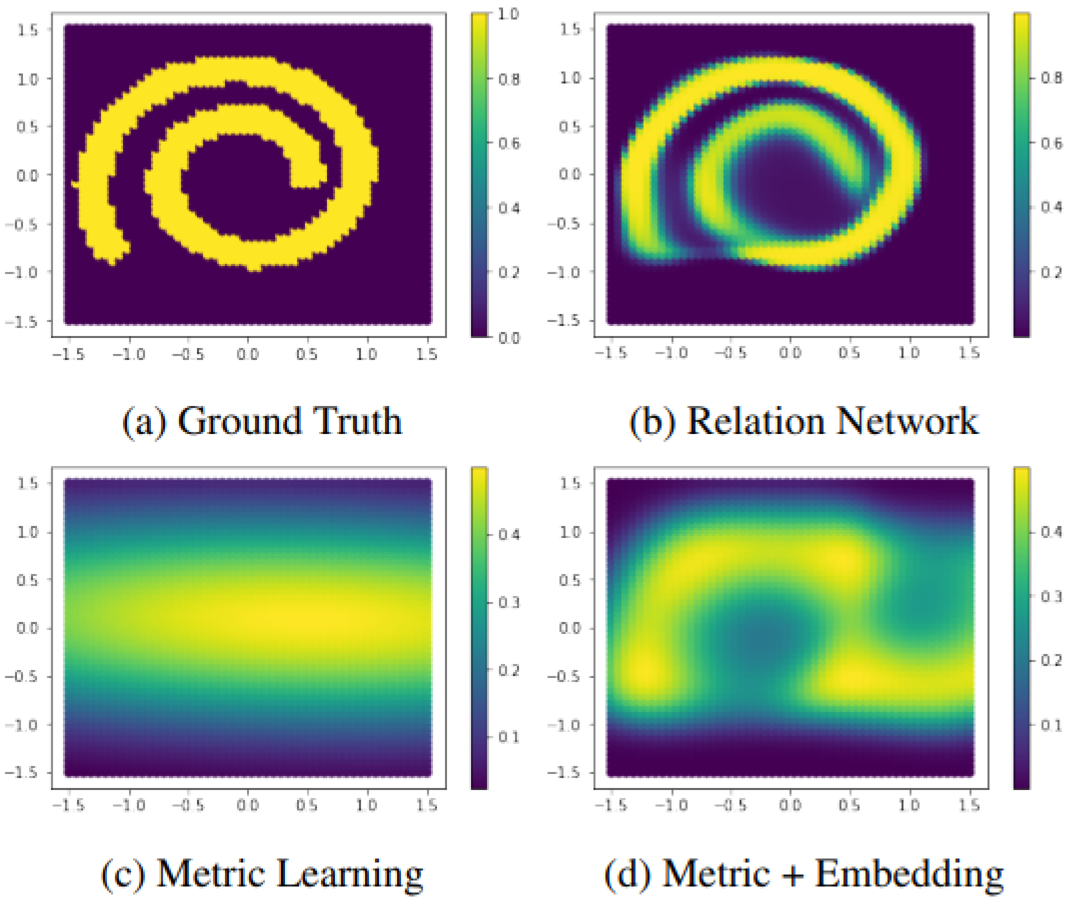

Figure 4: An example relation learnable by Relation Network and not by non-linear embedding + metric learning.

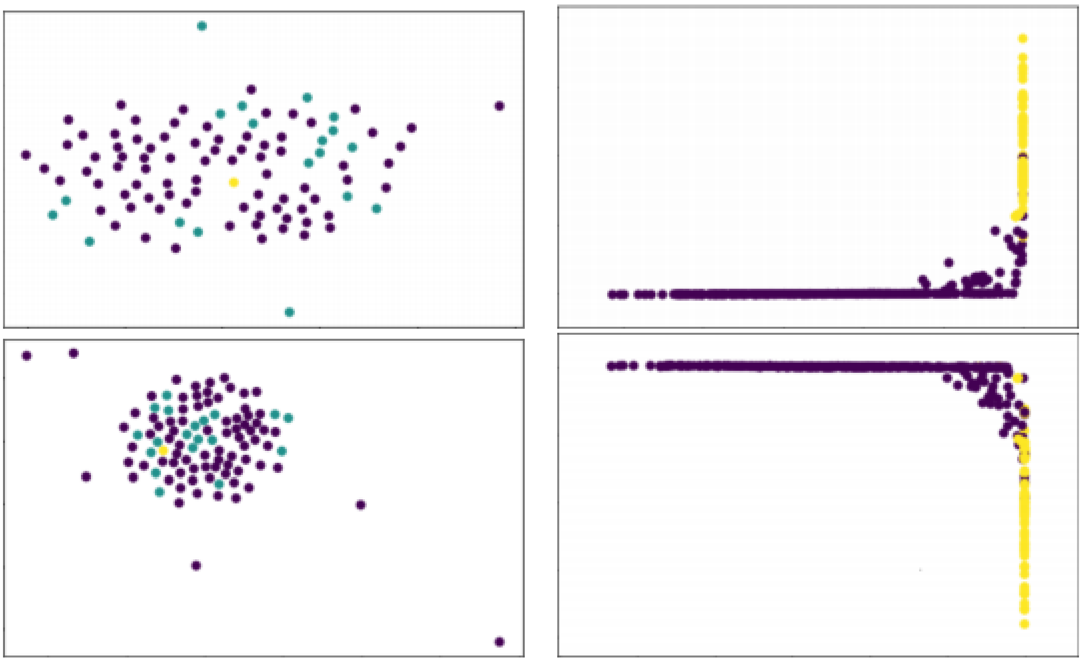

Figure 5: Example Omniglot few-shot problem visualisations. Left: Matched (cyan) and mismatched (magenta) sample embeddings for a given query (yellow) are not straightforward to differentiate. Right: Matched (yellow) and mismatched (magenta) relation module pair representations are linearly separable.