AMC
AI-ML.Club

Few-Shot Text Classification with Induction Network

基于小样本学习的意图识别冷启动

行习铭

Introduction : data deficiency problem

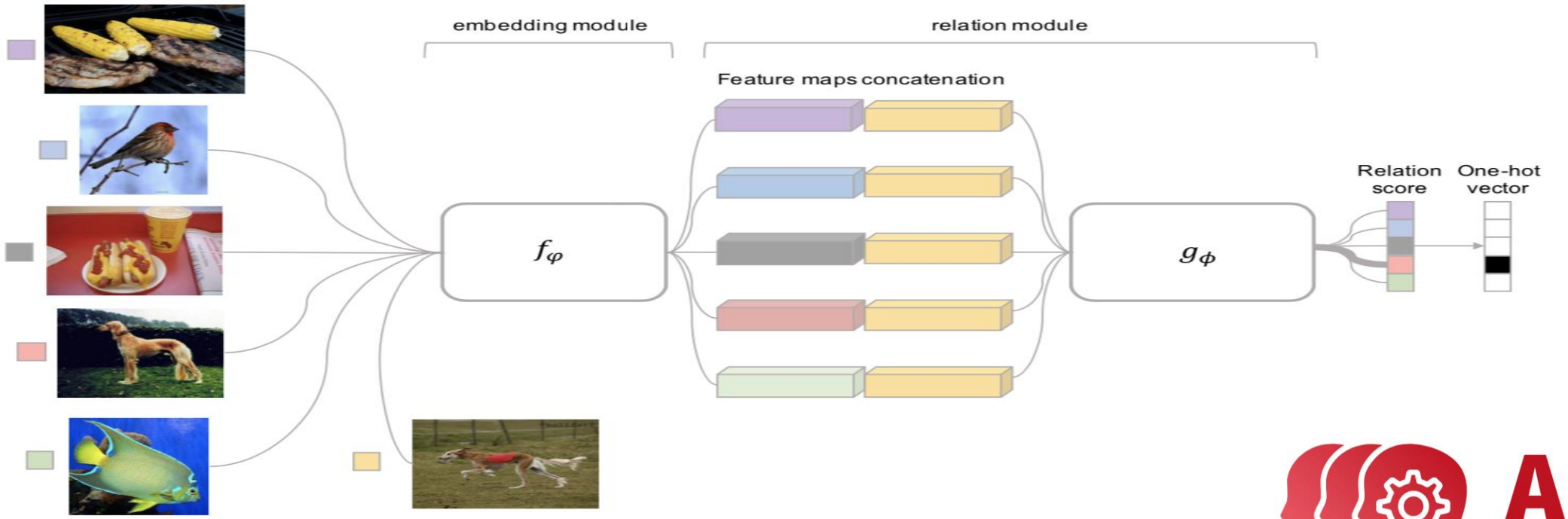
Deep learning has achieved a great success in many fields such as computer vision, speech recognition and natural language processing.

However, supervised deep learning is notoriously greedy for large labeled datasets, which limits the generalizability of deep models to new classes due to annotation cos.

AMC
AI-ML.Club

Introduction : C-way K-shot Problem

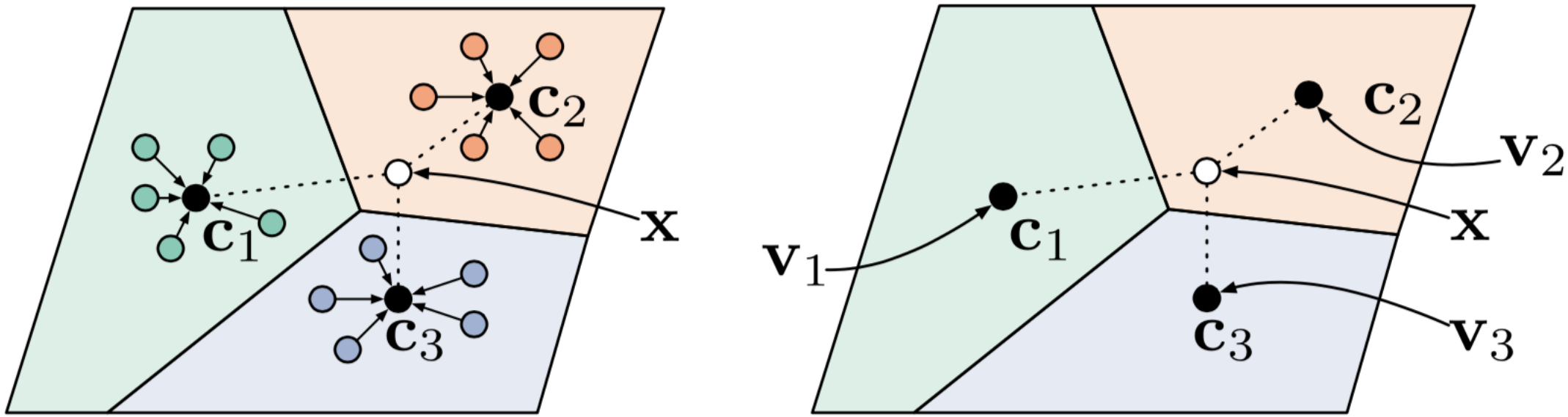
If the support set contains K labelled examples for each of C unique classes, the target few-shot problem is called C-way K-shot.



AMC
AI-ML.Club

Related work : Distance Metric Learning

The core idea in metric-based few- shot learning is similar to nearest neighbours and kernel density estimation.



(a) Few-shot

(b) Zero-shot

Problem Definition : Few-Shot Classification

Few-shot classification is a task in which a classifier must be adapted to accommodate new classes not seen in training, given only a few examples of each of these new classes.

We have a large labeled training set with a set of classes C_train. However, after training, our ultimate goal is to produce classifiers on the testing set with a disjoint set of new classes C_test, for which only a small labeled support set will be available.

Problem Definition : Training Procedure

Algorithm 1 Episode-Based Meta Training

1: **for** each *episode iteration* **do**
2: Randomly select C classes from the class space of the training set;
3: Randomly select K labeled samples from each of the C classes as support set $S = \{(x_s, y_s)\}_{s=1}^{m}$ $(m = K \times C)$, and select a fraction of the reminder of those C classes' samples as query set $Q = \{(x_q, y_q)\}_{q=1}^{n}$;
4: Feed the support set S to the model and update the parameters by minimizing the loss in the query set Q;
5: **end for**

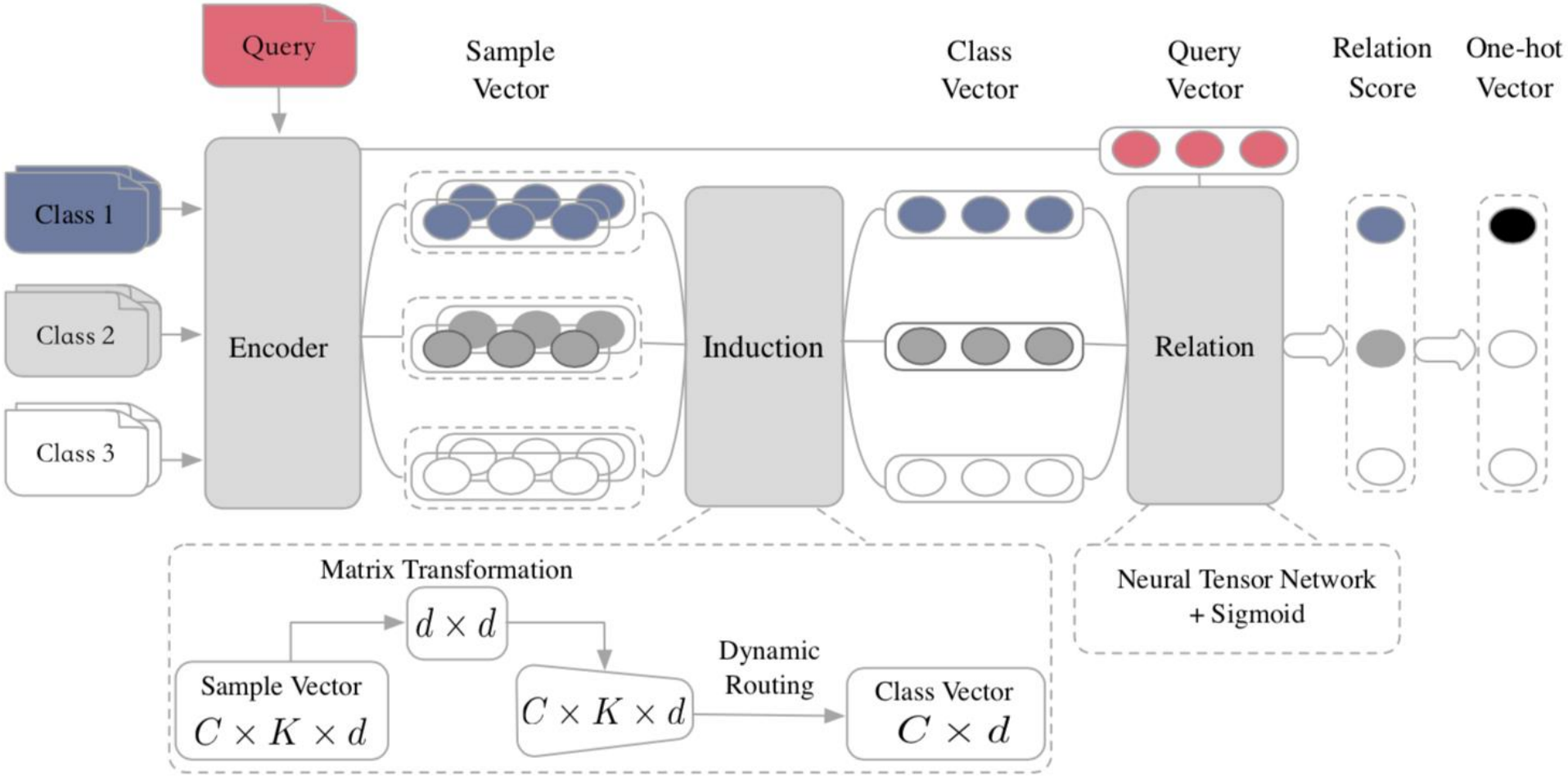
For example, if a dataset contains 159 training classes, this leads to

$$\begin{pmatrix} 159 \\ 5 \end{pmatrix} = 794,747,031$$

possible $5-$ way tasks.

AMC
AI-ML.Club

The Models : Overall structure



The Models : Encoder Module

This module is a bi-direction recurrent neural network with self-attention.

We use a bidirectional LSTM to process the text:

$$\overrightarrow{h_t} = \overrightarrow{LSTM}(w_t, h_{t-1}) \quad (1)$$

$$\overleftarrow{h_t} = \overleftarrow{LSTM}(w_t, h_{t+1}) \quad (2)$$

Computing the linear combination requires the self-attention mechanism:

$$a = softmax(W_{a2} tanh(W_{a1} H^T)) \quad (3)$$

The final representation e of the text is the weighted sum of H:

$$e = \sum_{t=1}^{T} a_t \cdot h_t \quad (4)$$

The Models : Induction Module

The main purpose of the induction module is to design a non-linear mapping from sample vector e_{sij} to class vector c_i:

$$\left\{e_{ij}^{s} \in R^{2u}\right\}_{i=1,\ldots C, j=1\ldots K} \mapsto \left\{c_i \in R^{2u}\right\}_{i=1}^{C}.$$

Algorithm 2 Dynamic Routing Induction

Require: sample vector e_{ij}^{s} in support set S and initialize the logits of coupling coefficients $b_{ij} = 0$

Ensure: class vector c_i

for all samples $j = 1, ..., K$ in class i:
  $\hat{e}_{ij}^{s} = squash(W_s e_{ij}^{s} + b_s)$
for $iter$ iterations **do**
    $d_i = softmax\,(b_i)$
    $\hat{c}_i = \sum_j d_{ij} \cdot \hat{e}_{ij}^{s}$
    $c_i = squash(\hat{c}_i)$
    for all samples $j = 1, ..., K$ in class i:
      $b_{ij} = b_{ij} + \hat{e}_{ij}^{s} \cdot c_i$
end for
Return c_i

Further Analysis : Effect of Transformation

(a) Before transformation (b) After transformation

AMC
AI-ML.Club

Further Analysis : Query Text Vector Visualization

(a) Relation Network (b) Induction Networks

The Models : Relation Module

The final relation score r_{iq} between the i-th class and the q-th query is calculated by a fully connected layer activated by a sigmoid function.

$$v(c_i, e^q) = f\left({c_i}^T M^{[1:h]} e^q\right) \tag{11}$$

$$r_{iq} = sigmoid(W_r v(c_i, e^q) + b_r) \tag{12}$$

The Models : Objective Function

We use the mean square error (MSE) loss to train our model, regressing the relation score r_{iq} to the ground truth y_q: matched pairs have similarity 1 and the mismatched pair have similarity 0. Given the support set S with C classes and query set $Q = \{(x_q, y_q)\}_{q=1}^{n}$ in an episode, the loss function is defined as:

$$L(S, Q) = \sum_{i=1}^{C} \sum_{q=1}^{n} (r_{iq} - \mathbf{1}(y_q == i))^2 \quad (13)$$

Experiments

- Open Domain Intent Classification for Dialog System (ODIC)

	Training Set	Testing Set
Class Num	159	57
Data Num	195,775	2,279
Data Num/Class	≥ 77	$20 \sim 77$

Table 1: Details of ODIC

Model	Mean Acc
Matching Networks (Vinyals et al., 2016)	65.73
Prototypical Networks (Snell et al., 2017)	68.17
Graph Network (Garcia and Bruna, 2017)	82.61
Relation Network (Sung et al., 2018)	83.07
SNAIL (Mishra et al., 2018)	82.57
ROBUSTTC-FSL (Yu et al., 2018)	83.12
Induction Networks (ours)	**85.63**

Table 2: Comparison of mean accuracy (%) on ARSC

Experiments

- Amazon Review Sentiment Classification (ARSC)

Model	5-way Acc.		10-way Acc.	
	5-shot	10-shot	5-shot	10-shot
Matching Networks (Vinyals et al., 2016)	82.54±0.12	84.63±0.08	73.64±0.15	76.72±0.07
Prototypical Networks (Snell et al., 2017)	81.82±0.08	85.83±0.06	73.31±0.14	75.97±0.11
Graph Network (Garcia and Bruna, 2017)	84.15±0.16	87.24±0.09	75.58±0.12	78.27±0.10
Relation Network (Sung et al., 2018)	84.41±0.14	86.93±0.15	75.28±0.13	78.61±0.06
SNAIL (Mishra et al., 2018)	84.62±0.16	87.31±0.11	75.74±0.07	79.26±0.09
Induction Networks (ours)	**87.16**±0.09	**88.49**±0.17	**78.27**±0.14	**81.64**±0.08

Table 3: Comparison of mean accuracy (%) on ODIC

Code

- https://github.com/laohur/LearnToCompareText