# Meta-Information Guided Meta-Learning for Few-Shot Relation Classification

**Sanghyun Seo**

**May 24, 2021**

**Department of Computer Engineering at Dongguk University**

**Artificial Intelligence Laboratory**



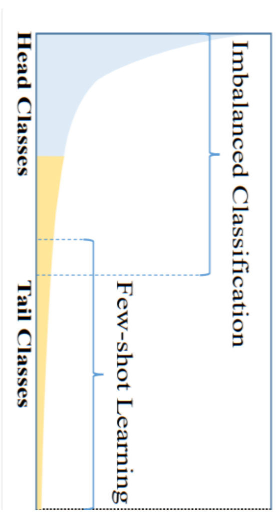
shseo@dongguk.edu

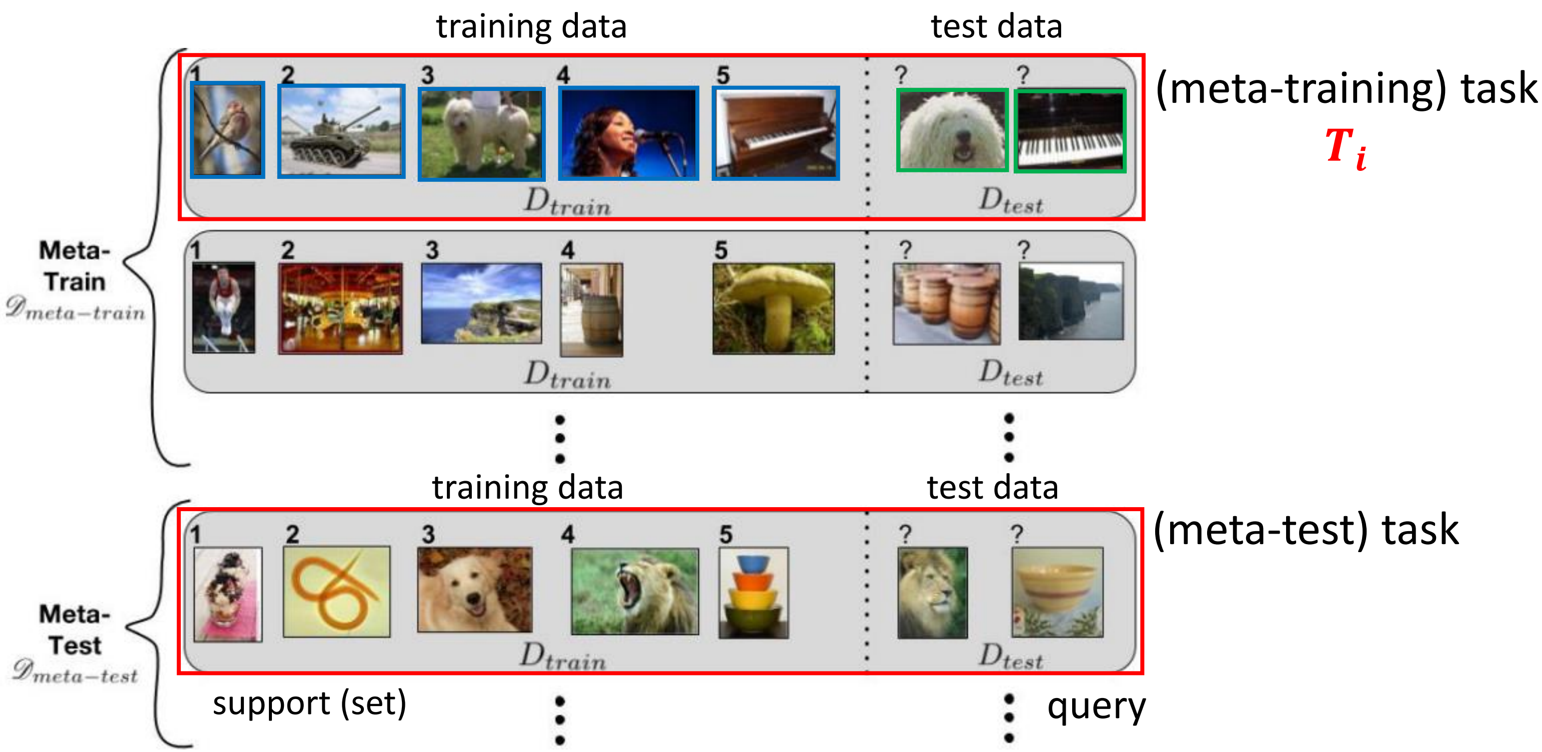
# Papers

- **Meta-Information Guided Meta-Learning for Few-Shot Relation Classification**
  - Dong, Bowen, et al. Proceedings of the 28th International Conference on Computational Linguistics. 2020.

# FSL Setting

- Meta-learning setup
  - $(D_{train}$ / $D_{test})$ / $(D_{train}$ / $D_{test})$ ← Meta-Train / Meta-Test
  - $\boldsymbol{D_i^{tr}} = \{(x_1^i, y_1^i), \dots, (x_k^i, y_k^i)\}, \boldsymbol{D_i^{ts}} = \{(x_1^i, y_1^i), \dots, (x_l^i, y_l^i)\}$
  - Task(episode) $\boldsymbol{T_i} = \{D_i^{tr}, D_i^{ts}\}$




Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." (2016).

# Tasks in few-shot NLP

- Domain as task
  - **ARSC:** multi-domain sentiment classification
    - 23 domains, 3 binary classification tasks
      → total 69 tasks (12 tasks, 4 domains are target tasks)
  - **CNICN150:** multi-domain intent classification
    - 10 domains, 15 intents (total 150 intents)
      → 22,500 labeled example, 1200 out-of-scope instances)

- Class as task
  - **FewRel:** few-shot relation classification
    - 100 relations (tr:64/dev:16/te:20), same domain(Wikipedia corpus and Wikidata knowledge bases)
      → FewRel 2.0 added a new domain of test set and 'none-of-above' relation
  - **SNIPS:** few-shot intent classification
    - 7 intents (tr:5/te:2)

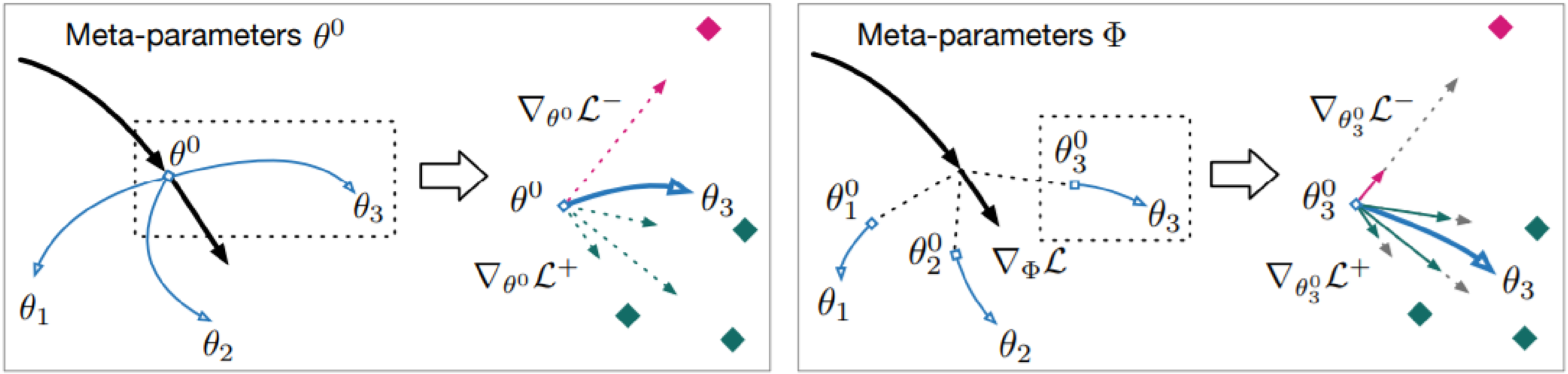
Yin, Wenpeng. "Meta-learning for few-shot natural language processing: A survey." arXiv preprint arXiv:2007.09604 (2020).

# Background

- Challenges of meta learning
  - Using only support set to classify query set
    - Most meta-learning methods learn how to learn (i.e., how to initialize and adapt) **solely relying on instance statistics**, which inevitably suffer from data sparsity and noise in low-resource scenarios, especially in text domain
  - Lack of interpretability
    - The approach of learning to learn, like the learning process itself, is a black-box and thus **lacks interpretability**
  - Weakness of zero-shot learning
    - Most conventional meta-learning methods are designed for few-shot classification, and **cannot well handle zero-shot scenarios**, where no support instances are available

# Methodology

- MIML (Meta Information guided Meta Learning)
  1) Instance encoder
  2) Meta-information guided fast initialization
  3) Meta-information guided fast adaptation
  4) Meta-optimization




(a) MAML (b) MIML

Figure 1: Diagram of meta-learning models. (a) MAML learns a class-agnostic representation $\theta^0$ that can fast adapt to new classes. (b) MIML learns meta-parameters $\Phi$ to fast initialize class-aware parameter $\theta_i^0$, and to quickly adapt to new classes using informative instances, where both phases are guided by meta-information. **Informative instances** and **noisy instances** are marked accordingly.

# Methodology

- Instance encoder
  - BERT model to encode the instance into contextualized representations

$$\mathrm{x}_j = g(x_j, h, t; \phi_e)$$

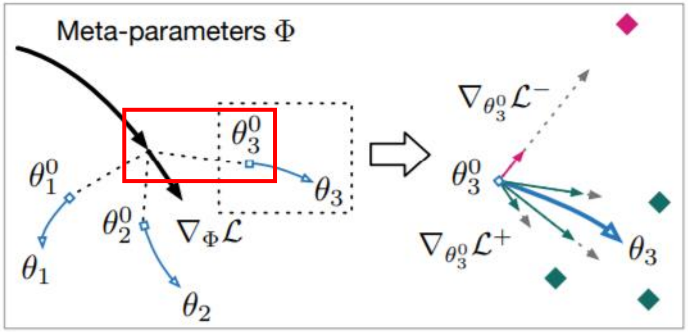
  - $x_j$ is the sentence, $h$ and $t$ are head and tail entities respectively. $g(\cdot)$ is the encoder, $\phi_e$ is the parameters of the encoder, and $\mathrm{x}_j \in \mathbb{R}^{d_s}$ is the instance representation

# Methodology

- Meta-information guided fast initialization
  - Instead of using a static class-agnostic initialization point for all classes as in MAML, MIML uses meta-information to estimate **dynamic class-aware initialization parameters** for each class
  - This alleviates the reliance on support instances to reach optimal adapted parameters

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
  $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
  Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
  Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
  **for all** $\mathcal{C}_i$ **do**
    Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
  **for** $t = 1, \ldots, T$ **do**
    Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
    Compute adapted parameters with gradient descent:
    $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t, \{\phi_e, \phi_n\}}, x_j, y_j)$
  Meta-optimize using query instance set $\mathcal{Q}$:
  $\Phi = \Phi - \beta \nabla_\Phi \mathcal{L}(f_{\theta^T, \Phi}, x_j, y_j)$




(b) MIML

# Methodology

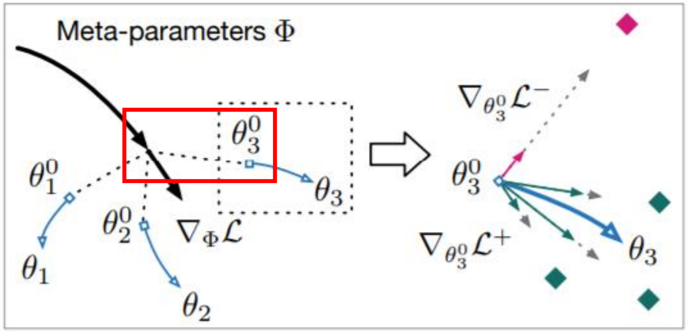
- Meta-information guided fast initialization
  - Given the name of a class $C_i$, the meta-information representation $c_i \in \mathbb{R}^{d_w}$ is obtained by **the average of the word embeddings of the name**

$$\theta_i^0 = \Psi(c_i; \phi_n)$$

  - where $\theta_i^0 \in \mathbb{R}^{d_s}$ is the class-aware initialization parameters for class $C_i$ , $\Psi(c_i; \phi_n)$ is the **meta-initializer**, $\phi_n$ is the corresponding meta-parameters

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
  $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
  Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
  Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
  **for all** $\mathcal{C}_i$ **do**
    Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
  **for** $t = 1, \ldots, T$ **do**
    Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
    Compute adapted parameters with gradient descent:
    $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t,\{\phi_e,\phi_n\}}, x_j, y_j)$
  Meta-optimize using query instance set $\mathcal{Q}$:
  $\Phi = \Phi - \beta \nabla_{\Phi} \mathcal{L}(f_{\theta^T,\Phi}, x_j, y_j)$




(b) MIML

# Methodology

- Meta-information guided fast initialization
  - $\Psi(\cdot)$ is implemented via a fully connected layer
  - It usually is a rough in an early stage, but flexible estimation of a new concept based on its high-level semantics is possible

$$s_{i,j} = {\theta_i^0}^T x_j$$

  - where $s_{i,j}$ is the score of $x_j$ being an instance of $C_i$. The probability $p(y = C_i | x_j)$ is obtained by normalizing the score $s_{i,j}$ with a softmax layer over all classes $\{C_1, C_2, \ldots, C_N\}$
  - The model after fast initialization can be denoted as $f_{\theta_0,\{\phi_e,\phi_n\}}$, where $\theta^0 = \{\theta_1^0, \theta_2^0, \ldots, \theta_N^0\}$ denotes initialized parameters

# Methodology

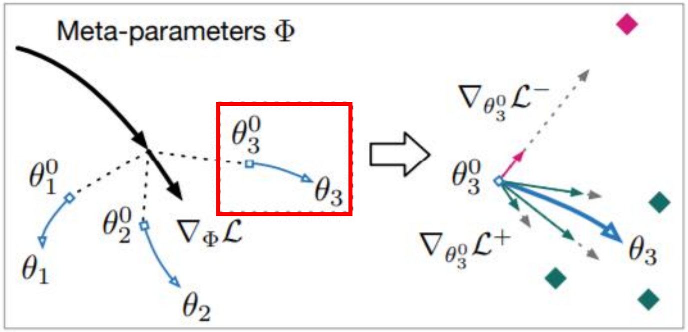
- Meta-information guided fast adaptation
  - The initialized parameters $\theta^0$ are adapted via gradient descent steps according to the classification performance of instances on the support set $S$
  - The adaptation iterates dynamically for $T$ steps

$$\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta_0,\{\phi_e,\phi_n\}}, x_j, y_j)$$

  - $\mathcal{L}(\cdot)$ denotes cross-entropy loss of a support instance

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
    $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
    Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
    Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
    **for all** $\mathcal{C}_i$ **do**
        Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
    **for** $t = 1, \ldots, T$ **do**
        Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
        Compute adapted parameters with gradient descent:
        $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t,\{\phi_e,\phi_n\}}, x_j, y_j)$
    Meta-optimize using query instance set $\mathcal{Q}$:
    $\Phi = \Phi - \beta \nabla_\Phi \mathcal{L}(f_{\theta^T,\Phi}, x_j, y_j)$




(b) MIML

# Methodology

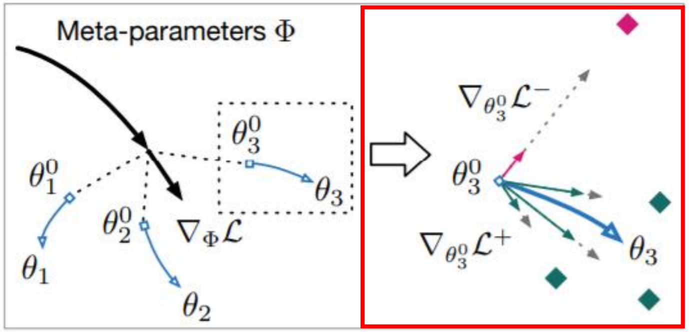
- Meta-information guided fast adaptation
  - To select informative instances for fast adaptation in MIML, instead of using a static learning rate for all instances, the learning rate of each instance is dynamically determined by a selective attention mechanism as follows:

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_j \exp(e_{i,j})}$$

  - where $e_{i,j}$ is the score of instance $x_j$ for class $C_i$.

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
  $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
  Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
  Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
  **for all** $\mathcal{C}_i$ **do**
    Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
  **for** $t = 1, \ldots, T$ **do**
    Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
    Compute adapted parameters with gradient descent:
    $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t, \{\phi_e, \phi_n\}}, x_j, y_j)$
  Meta-optimize using query instance set $\mathcal{Q}$:
  $\Phi = \Phi - \beta \nabla_\Phi \mathcal{L}(f_{\theta^T, \Phi}, x_j, y_j)$




(b) MIML

# Methodology

- Meta-information guided fast adaptation
  - The score is obtained by:

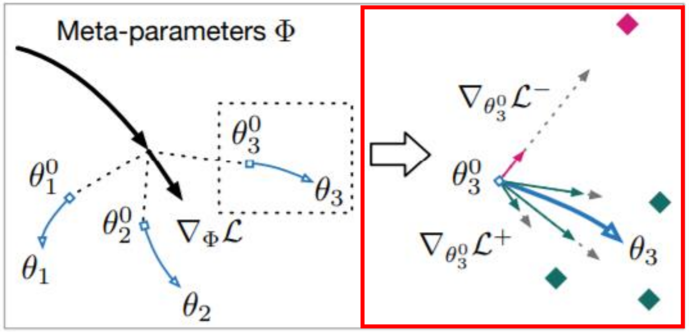
$$e_{i,j} = {q_i}^T x_j$$

  - Where $q_i \in \mathbb{R}^{d_s}$ is the query vector for class $C_i$
  - Estimating the query vector from meta-information via a meta-querier module as follows:

$$q_i = \Psi(c_i; \phi_a)$$

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
    $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
    Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
    Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
    **for all** $\mathcal{C}_i$ **do**
        Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
        **for** $t = 1, \ldots, T$ **do**
            Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
            Compute adapted parameters with gradient descent: $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t, \{\phi_e, \phi_n\}}, x_j, y_j)$
    Meta-optimize using query instance set $\mathcal{Q}$:
    $\Phi = \Phi - \beta \nabla_\Phi \mathcal{L}(f_{\theta^T, \Phi}, x_j, y_j)$




(b) MIML

# Methodology

- Meta-information guided fast adaptation
  - The score is obtained by:

$$e_{i,j} = {q_i}^T x_j$$

  - where $q_i \in \mathbb{R}^{d_s}$ is the query vector for class $C_i$

  - The estimated query vector from meta-information via a **meta-querier module** as follows:

$$q_i = \Psi(c_i; \phi_a)$$

- Overfitting Problem
  - **L2 normalization**
  - Virtual adversarial training

# Methodology

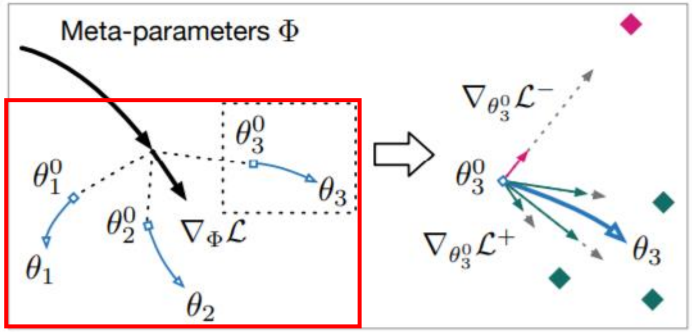
- Meta-optimization
  - After fast adaptation on support instances, the meta-parameters $\Phi = \{\phi_e, \phi_n, \phi_a\}$ are optimized according to the performance of the adapted model on the query set $Q$ as follows:

$$\Phi = \Phi - \beta \nabla_{\Phi} \mathcal{L}(f_{\theta^T,\Phi}, x_j, y_j)$$

  - where β is the learning rate for meta-parameters

**Algorithm 1** Meta-Information Guided Meta-Learning

**Require:** $p(\mathcal{C})$: distribution over classes
**Require:** $\beta$: meta learning rate
randomly initialize:
  $\Phi = \{\phi_e, \phi_n, \phi_a\}$: meta-parameters
**while** not done **do**
  Sample batch of classes $\mathcal{C}_i \sim p(\mathcal{C})$
  Sample support instance set $\mathcal{S}$ and query instance set $\mathcal{Q}$
  **for all** $\mathcal{C}_i$ **do**
    Fast initialize parameters of $\mathcal{C}_i$: $\theta_i^0 = \Psi(\mathbf{c}_i; \phi_n)$
  **for** $t = 1, \ldots, T$ **do**
    Compute gradients and learning rates for fast adaptation using support instance set $\mathcal{S}$
    Compute adapted parameters with gradient descent:
    $\theta^{t+1} = \theta^t - \sum_{i,j} \alpha_{i,j} \nabla_{\theta^t} \mathcal{L}(f_{\theta^t,\{\phi_e,\phi_n\}}, x_j, y_j)$
  Meta-optimize using query instance set $\mathcal{Q}$:
  $\Phi = \Phi - \beta \nabla_{\Phi} \mathcal{L}(f_{\theta^T,\Phi}, x_j, y_j)$




(b) MIML

# Methodology

- Implementation Details
  - Model: $BERT_{base}$ with GloVe 50d word embeddings
  - class distribution $p(C)$: uniform distribution
  - # of adaptation step: 150
  - optimizer: Adam

- Dataset & Evaluation Protocol
  - Dataset: FewRel(70,000 labeled sentences in 100 relations)
  - Evaluation: 5-way 1-shot, 5-way 5-shot, 10-way 1-shot, 10-way 5-shot.
  - Baseline: MetaNets, GNN, SNAIL, ProtoNets, MLMAN, BERT-PAIR, ProtoNets(with BERT encoder), MAML(with BERT encoder)

# Experiments

- Main results
  - Meta-information guided fast initialization in MIML can produce more flexible class-aware initialization, which alleviates heavy reliance on support instances

| Encoder | Model | 5-way-1-shot | 5-way-5-shot | 10-way-1-shot | 10-way-5-shot |
|---|---|---|---|---|---|
| CNN | Meta Network* | $64.46 \pm 0.54$ | $80.57 \pm 0.48$ | $53.96 \pm 0.56$ | $69.23 \pm 0.52$ |
| | GNN* | $66.23 \pm 0.75$ | $81.28 \pm 0.62$ | $46.27 \pm 0.80$ | $64.02 \pm 0.77$ |
| | SNAIL* | $67.29 \pm 0.26$ | $79.40 \pm 0.22$ | $53.28 \pm 0.27$ | $68.33 \pm 0.25$ |
| | Proto Network* | $74.52 \pm 0.07$ | $88.40 \pm 0.06$ | $62.38 \pm 0.06$ | $80.45 \pm 0.08$ |
| | MLMAN* | $82.98 \pm 0.20$ | $92.66 \pm 0.09$ | $73.59 \pm 0.26$ | $87.29 \pm 0.15$ |
| BERT | BERT-PAIR ♠ | $88.32 \pm 0.64$ | $93.22 \pm 0.13$ | $80.63 \pm 0.17$ | $87.02 \pm 0.12$ |
| | MAML | $87.45 \pm 0.11$ | $94.39 \pm 0.13$ | $78.91 \pm 0.14$ | $89.14 \pm 0.23$ |
| | Proto Network | $86.50 \pm 0.14$ | $95.01 \pm 0.15$ | $82.86 \pm 0.15$ | $91.30 \pm 0.11$ |
| | MIML | $\mathbf{92.55 \pm 0.12}$ | $\mathbf{96.03 \pm 0.17}$ | $\mathbf{87.47 \pm 0.21}$ | $\mathbf{93.22 \pm 0.22}$ |
| - | Human* | 92.22 | - | 85.88 | - |

Table 1: Main results. Accuracies (%) on few-shot relation classification on FewRel test set. Results with * and ♠ are from FewRel leaderboard and Gao et al. (2019b) respectively.

# Experiments

- Robustness to Noisy Instances
  - Randomly corrupt 0%, 10%, 20%, 30% support instances, by replacing them with noisy instances randomly sampled from different relations in FewRel

| Model | Noise Rate | 5-way-5-shot | 10-way-5-shot | Noise Rate | 5-way-5-shot | 10-way-5-shot |
|---|---|---|---|---|---|---|
| MAML | 0% | $92.59 \pm 0.08$ | $85.79 \pm 0.15$ | 10% | $90.81 \pm 0.12$ | $83.31 \pm 0.13$ |
| Proto Network | | $92.62 \pm 0.11$ | $87.12 \pm 0.12$ | | $91.54 \pm 0.08$ | $85.40 \pm 0.18$ |
| Proto HATT | | $93.43 \pm 0.09$ | $89.37 \pm 0.17$ | | $92.40 \pm 0.13$ | $88.19 \pm 0.22$ |
| MIML | | $\mathbf{95.60 \pm 0.09}$ | $\mathbf{91.60 \pm 0.21}$ | | $\mathbf{94.82 \pm 0.08}$ | $\mathbf{89.55 \pm 0.25}$ |
| MAML | 20% | $88.40 \pm 0.10$ | $80.77 \pm 0.13$ | 30% | $86.18 \pm 0.20$ | $78.30 \pm 0.11$ |
| Proto Network | | $91.04 \pm 0.08$ | $83.18 \pm 0.17$ | | $87.84 \pm 0.12$ | $80.28 \pm 0.19$ |
| Proto HATT | | $91.27 \pm 0.15$ | $85.94 \pm 0.29$ | | $89.62 \pm 0.19$ | $83.14 \pm 0.24$ |
| MIML | | $\mathbf{93.19 \pm 0.10}$ | $\mathbf{87.70 \pm 0.23}$ | | $\mathbf{92.04 \pm 0.18}$ | $\mathbf{86.19 \pm 0.27}$ |

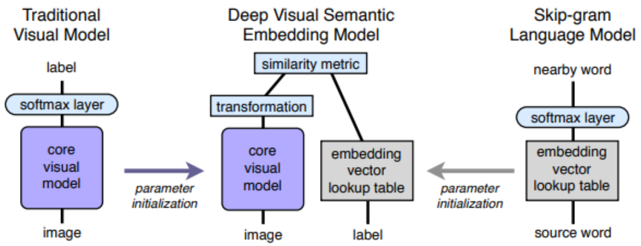
Table 2: Accuracies (%) on few-shot relation classification with noise on FewRel development set.

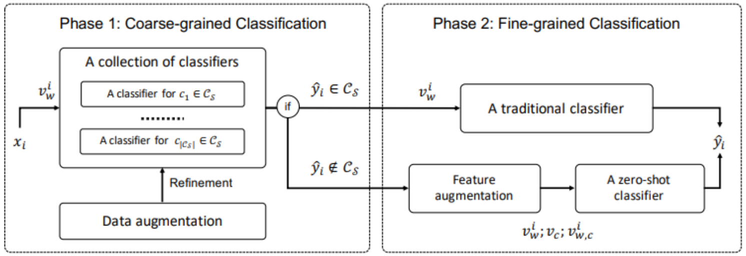
# Experiments

- Zero-Shot Classification
  - Remove the support instances in evaluation phase in 5-way and 10-way setting, and ask the model to classify query instances with class-aware initialization parameters
    - DeViSE model with BERT encoder
    - SK4 with rich semantic knowledge of classes, including word embeddings, class descriptions, class hierarchy, and commonsense knowledge graphs

| Setting | Random | DeViSE | SK4 | MIML |
|---|---|---|---|---|
| 5-way-0-shot | 20.00 | $55.90 \pm 0.09$ | $\mathbf{79.68 \pm 0.12}$ | $79.54 \pm 0.06$ |
| 10-way-0-shot | 10.00 | $42.29 \pm 0.08$ | $\mathbf{66.17 \pm 0.11}$ | $61.14 \pm 0.10$ |

Table 3: Experimental results of zero-shot classification on FewRel development set.






Artificial Intelligence Laboratory
Department of Computer Engineering at Dongguk University

# Experiments

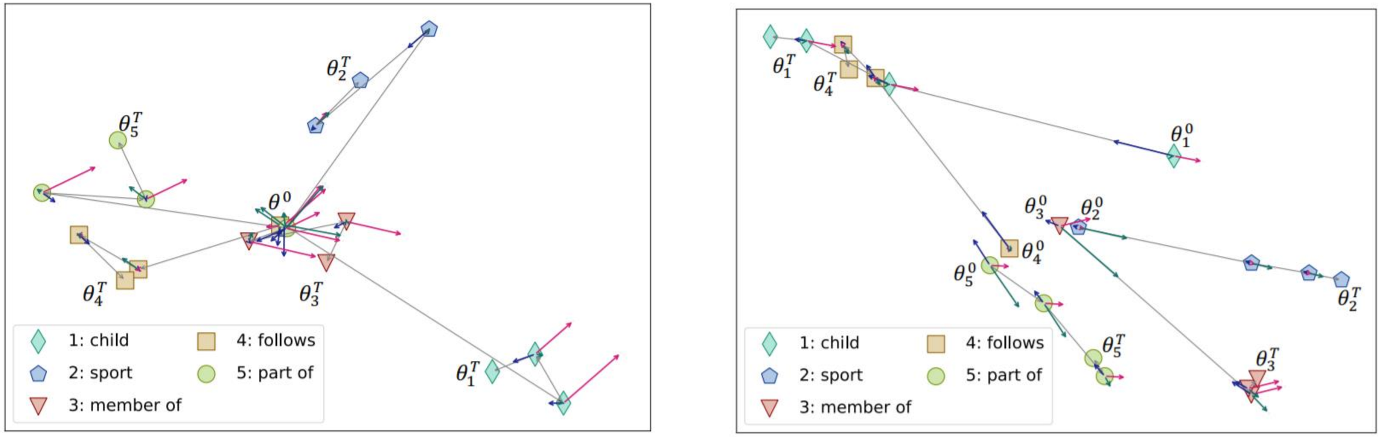
- Ablation Study
  - Ablation study in 10-way5-shot setting, by removing each component, including meta-information guided fast initialization (MI) and adaptation (MA), class-aware parameter normalization (NM) and virtual adversarial training (VAT)

| Model | MAML | MIML | MIML w/o MI | MIML w/o MA | MIML w/o NM | MIML w/o VAT |
|---|---|---|---|---|---|---|
| Accuracy | $85.79 \pm 0.15$ | $\mathbf{91.60 \pm 0.21}$ | $86.43 \pm 0.17$ | $89.59 \pm 0.19$ | $84.17 \pm 0.13$ | $89.43 \pm 0.09$ |

Table 4: Ablation results in 10-way-5-shot setting on FewRel development set. MI/MA: meta-information guided fast initialization/adaptation, NM: Normalization, VAT: virtual adversarial training.

# Experiments

- Visualization
  - Visualizing the workflow of MIML in the presence of 20% noise in 5-way-5-shot setting and comparing it with MAML
  - The initialization representations and adaptation steps are visualized by applying principal component analysis




(a) MAML (b) MIML

Figure 2: Visualization of initialization and adaptation process of meta-learning models, in 5-way-5-shot setting with 20% noise. At each iteration, the adaptation gradients for a class parameter $\theta_i$ come from three parts: informative instances from class $C_i$ (marked in **green** arrows), noisy instance for class $C_i$ (marked in **red** arrows), and instances for other classes (marked in **blue** arrows).[1] Best viewed in color.

# Future Works

- Meta information
  - Exploring more meta-information for meta-learning, such as class descriptions and knowledge graphs
- Enhanced Encoder
  - Developing more sophisticated models to capture the fine-grained interactions between the high-level meta information and concrete instances, to better guide meta-learning for few-shot classification problem
- Hybrid approach for meta learning
  - Integrating optimization-based approaches and metric-based approaches to make a better performance, to do few-shot classification and zero-shot classification simultaneously

# Q&A
# Thank you!