Learning Data Manipulation for Augmentation and Weighting

Zhiting Hu*, Bowen Tan*, Ruslan Salakhutdinov, Tom Mitchell, Eric Xing
Carnegie Mellon University, Petuum Inc.
Learning Data Manipulation

• Data manipulation is often a crucial step to improve performance

  • Data **augmentation** for **small data problems**
    
    \[ y = \text{cat} \quad x = \quad \text{augment} \quad x' = \]

    • Rule-based: image rotation, cropping; text synonym, ...
    • Learning-based [Ratner et al., 17; Cubuk et al., 19]

  • Data **weighting** for **class-imbalance problems**
    
    \[ w = 1 \quad w = 10 \]

    • Rule-based: inverse class frequency, ...
    • Learning-based: meta-learning [Ren et al., 18]; self-paced learning [Jiang et al., 18]
Learning Data Manipulation

• Data manipulation is often a crucial step to improve performance

  • Data **augmentation** for **small data problems**
    
    \[ y = \text{cat} \quad x = \quad \text{augment} \quad x' = \]

    • Rule-based: image rotation, cropping; text synonym, ...
    • Learning-based [Ratner et al., 17; Cubuk et al., 19]

  • Data **weighting** for **class-imbalance problems**
    
    \[ w = 1 \quad w = 10 \]

    • Rule-based: inverse class frequency, ...
    • Learning-based: meta-learning [Ren et al., 18]; self-paced learning [Jiang et al., 18]

Can we have a more generic learning method?

Specific to one manipulation type
Background

Connecting the Dots between MLE and RL [Tan et al., 2019]
Parameterize Manipulation

- Standard MLE

$$\mathcal{L}_{MLE,SL}(q, \theta) = \mathbb{E}_{q(x,y)}[f_\delta(x, y|\mathcal{D})] - \alpha \text{KL}(q(x, y) || p(x)p_\theta(y|x)) + H(q)$$

$$f_\delta(x, y|\mathcal{D}) = \begin{cases} 1 & \text{if } (x, y) \in \mathcal{D} \\ -\infty & \text{otherwise} \end{cases}$$

Get valid rewards only when samples match training data
Parameterize Manipulation

- Standard MLE

\[
\mathcal{L}_{\text{MLE}, SL}(q, \theta) = \mathbb{E}_{q(x, y)}[f_\delta(x, y|\mathcal{D})] - \alpha \text{KL}(q(x, y)||p(x)p_\theta(y|x)) + H(q)
\]

\[
f_\delta(x, y|\mathcal{D}) = \begin{cases} 
1 & \text{if } (x, y) \in \mathcal{D} \\
-\infty & \text{otherwise}
\end{cases}
\]
Parameterize Manipulation

- Standard MLE
  \[ \mathcal{L}_{\text{MLE},SL}(q, \theta) = \mathbb{E}_{q(x,y)}[f_\delta(x,y|D)] - \alpha \text{KL}(q(x,y)||p(x)p_\theta(y|x)) + H(q) \]

  \[ f_\delta(x,y|D) = \begin{cases} 
  1 & \text{if } (x,y) \in D \\
  -\infty & \text{otherwise}
  \end{cases} \]

- Data augmentation
  \[ f_\phi^{\text{aug}}(x,y|D) = \begin{cases} 
  1 & \text{if } x \sim g_\phi(x|x^*,y) \quad (x^*,y) \in D \\
  -\infty & \text{otherwise}
  \end{cases} \]

- Data weighting
  \[ f_\phi^{w}(x,y|D) = \begin{cases} 
  \phi_i & \text{if } (x,y) = (x_i,y_i) \quad (x_i,y_i) \in D \\
  -\infty & \text{otherwise}
  \end{cases} \]

[Hu et al., NeurIPS2019]
Use a Reward Learning Algorithm

- **Intrinsic reward learning** [Zheng et al., 18]
  - Augment extrinsic reward for better performance
    \[ r^{ex+in}(x, y) = r^{ex}(x, y) + r^{in}_{\phi}(x, y) \]
  - Update model:
    \[ \theta' = \theta + \gamma \nabla_{\theta} L^{ex+in}(\theta, \phi) \]
  - Update reward:
    \[ \phi' = \phi + \gamma \nabla_{\phi} L^{ex}(\theta'(\phi)) \]

Train model with augmented reward

New model $\theta'$ gets higher extrinsic reward

[Hu et al., NeurIPS2019]
Use a Reward Learning Algorithm

- **Intrinsic reward learning** [Zheng et al., 18]
  - Augment extrinsic reward for better performance
    \[ r^{ex+in}(x, y) = r^{ex}(x, y) + r^{in}_\phi(x, y) \]
  - Update model:
    \[ \theta' = \theta + \gamma \nabla_{\theta} \mathcal{L}^{ex+in}(\theta, \phi) \]
  - Update reward:
    \[ \phi' = \phi + \gamma \nabla_{\phi} \mathcal{L}^{ex}(\theta'(\phi)) \]
- **Map to data manipulation**
  - Update model:
    \[ \theta' = \theta + \gamma \nabla_{\theta} \mathcal{L}^{manip}(\theta, \phi) \]
  - Update data:
    \[ \phi' = \phi + \gamma \nabla_{\phi} \mathcal{L}^{v}(\theta'(\phi)) \]

[Hu et al., NeurIPS2019]

Train model with augmented reward
New model $\theta'$ gets higher extrinsic reward
Train model on manipulated data
New model $\theta'$ gets better valid-set perf.
Parameterization of Text Data Augmentation

Augment data

\[ \mathbf{x} = \text{Food tasted nice, but the service was very disappointing.} \]

Random mask

\[ \mathbf{x} = \text{Food [Mask] [Mask], but the service was very disappointing.} \]

Raw data

\[ \mathbf{x} = \text{Food was good, but the service was very disappointing.} \]

Finetune BERT to condition on the label

\[ y = \text{neg} \]

[Hu et al., NeurIPS2019]
Low-Data Text Classification

Train size: 40 per class; Val size: 2

[Hu et al., NeurIPS2019]

Use a label-conditional BERT as the augmentation model $g_\phi(x|x^*, y)$
Class-Imbalance Image Classification

CIFAR10 binary image classification

Accuracy

Imbalance ratio

Base: ResNet
Ours (weighting)
Ren
Proportion

[Hu et al., NeurIPS2019]