Adversarial Multi-task Learning for Text Classification

Multi-task Learning

- 여러 Task를 두고 하나의 모델로 학습시키는 학습 방법
- 보통 overfitting을 방지하기 위해 regularization parameter를 추가하는데, 이보다 자연적으로 다양한 task를 수행하기 위해 추가된 regularization이 더 효과적일 때가 있음. 이때 multi-task learning이 매우 효과적임

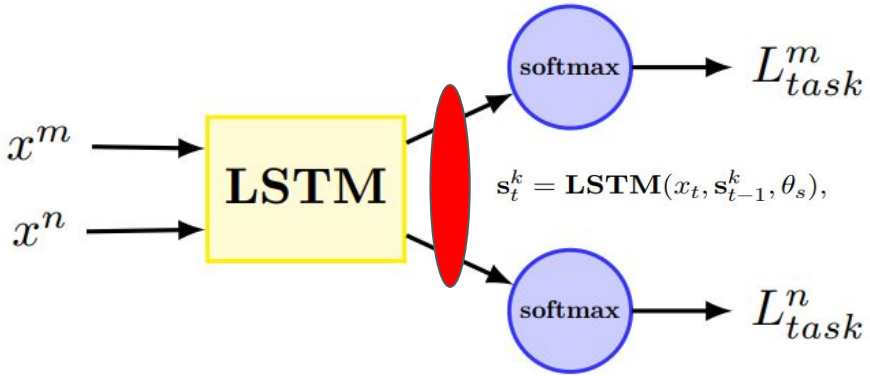
- Input - Label pair가 부족한 NLP에서 많이 사용됨

전이 학습과의 비교

- 전이학습은 최종적 task를 위해 그 전의 task들을 학습 시킴 (보통 한가지 task만 한다고 함)
  t(1), t(2), … , t(n-1) → t(n)
- Multi-task Learning은 포함된 모든 task를 동시에 학습 시킴
  t(1), t(2), … , t(n-1), t(n)
- Multi-task Learning을 전이학습의 범주에 포함시키는 경우도 있음
  Parallel Transfer Learning : Multi-task Learning
  Sequential Transfer Learning : Transfer Learning
- Multi-task Learning으로 학습된 모델도 전이 학습 모델로 사용될 수 있음

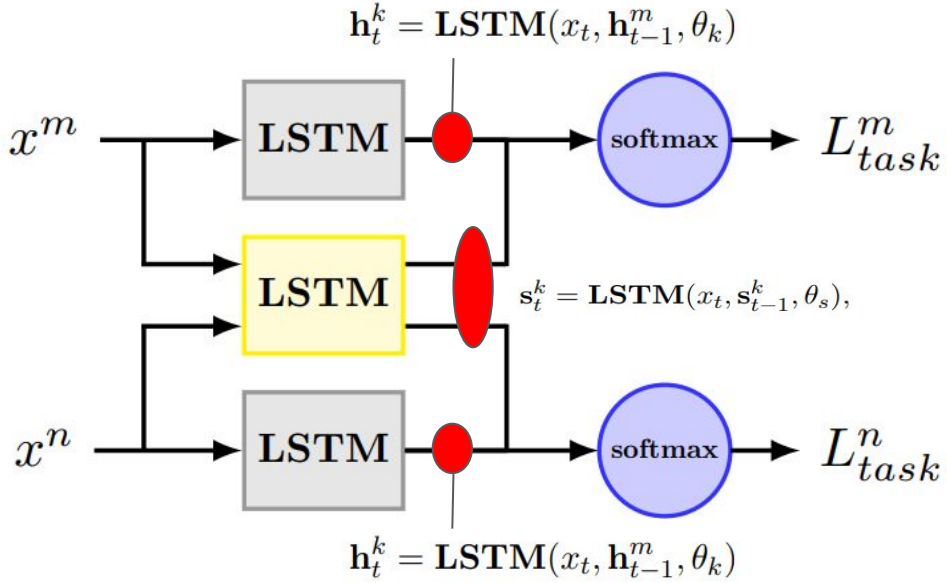
Multi-task Learning

1) Fully-Shared MTL



Multi-task Learning

2) Shared-Private MTL



Objective Function

$$L = L_{Task} + \lambda L_{Adv} + \gamma L_{Diff}$$

L_{Task}

Task-specific softmax

각 Task에 해당하는 Output Layer의 출력 값으로 Loss 값 계산

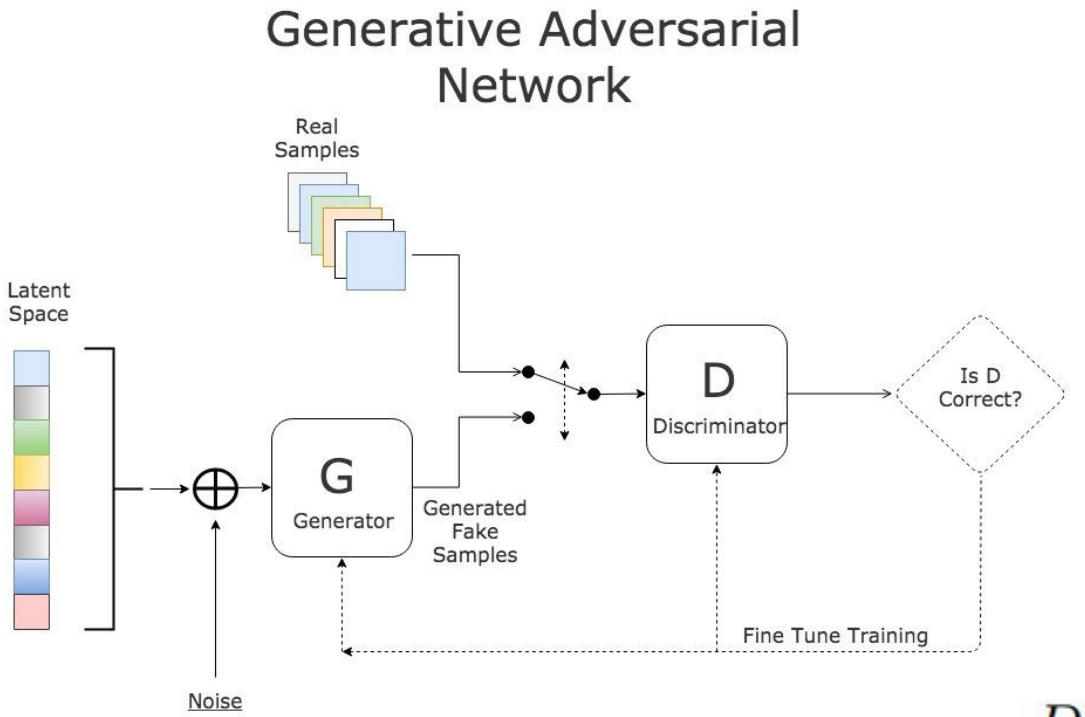
$$\mathbf{h}_t^k = \mathbf{LSTM}(x_t, \mathbf{h}_{t-1}^m, \theta_k)$$

Objective Function

$$L(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_i^j \log(\hat{y}_i^j),$$

$$L_{Task} = \sum_{k=1}^{K} \alpha_k L(\hat{y}^{(k)}, y^{(k)})$$

$$\lambda L_{Adv}$$



Real과 Fake 구분을
못하게 하는 것이 목적

Domain adaptation:
구분이 되지 않으면
transferable feature
= task-invariant feature

$$D(\mathbf{s}_T^k, \theta_D) = \text{softmax}(\mathbf{b} + \mathbf{U}\mathbf{s}_T^k)$$

λL_{Adv}

Adversarial Loss

Discriminator 모델이 구분을 못하게 Output 출력

$$\mathbf{s}_t^k = \mathbf{LSTM}(x_t, \mathbf{s}_{t-1}^k, \theta_s),$$

Objective function

$$\phi = \min_G \max_D \Big(E_{x \sim P_{data}}[\log D(x)] + E_{z \sim p(z)}[\log(1 - D(G(z)))] \Big)$$

$$L_{Adv} = \min_{\theta_s} \left(\lambda \max_{\theta_D} \left(\sum_{k=1}^{K} \sum_{i=1}^{N_k} d_i^k \log[D(E(\mathbf{x}^k))] \right) \right)$$

γL_{Diff}

Orthogonality Restriction

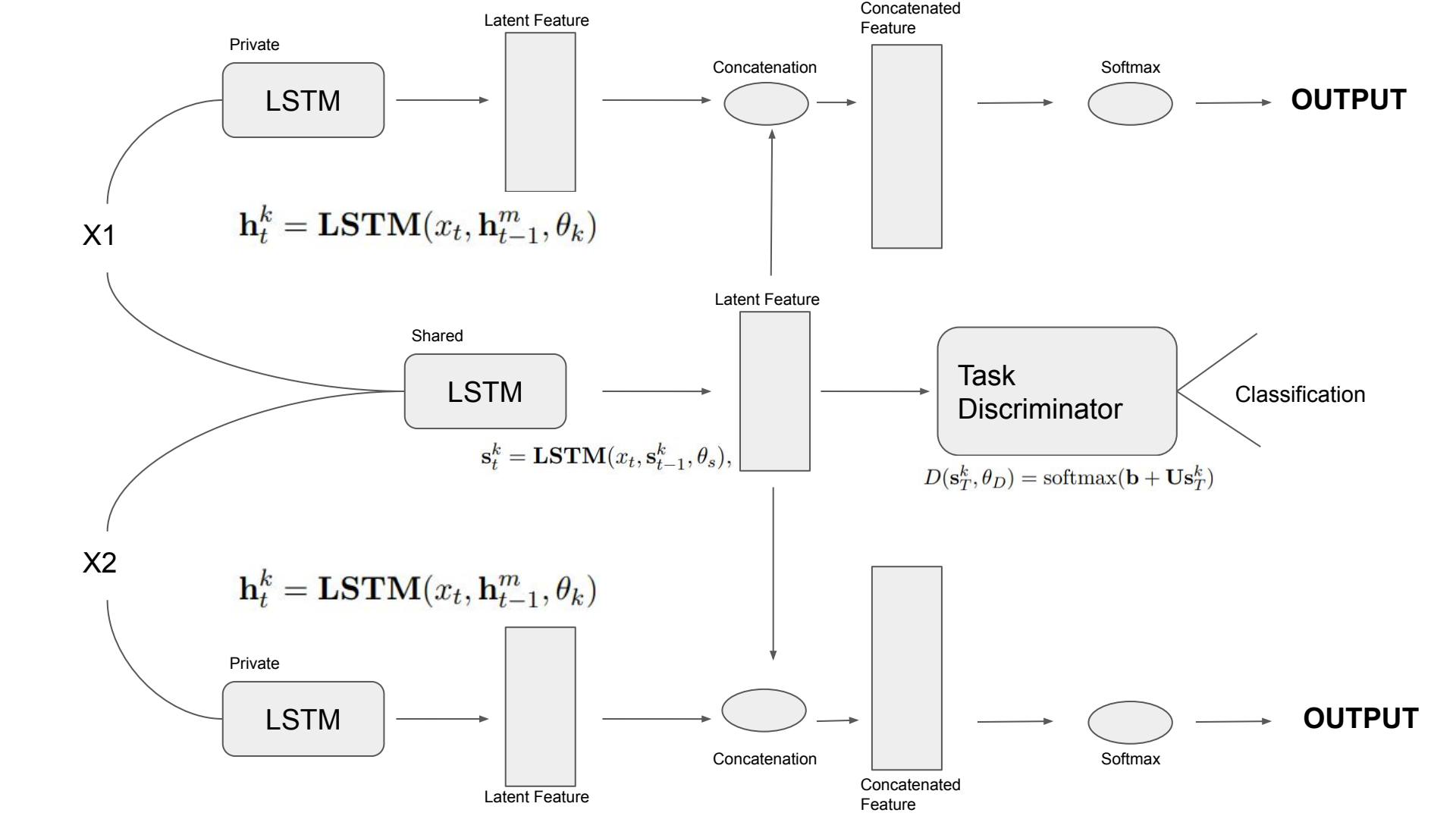
상호배타적 feature representation을 위해 추가 loss 함수를 더함

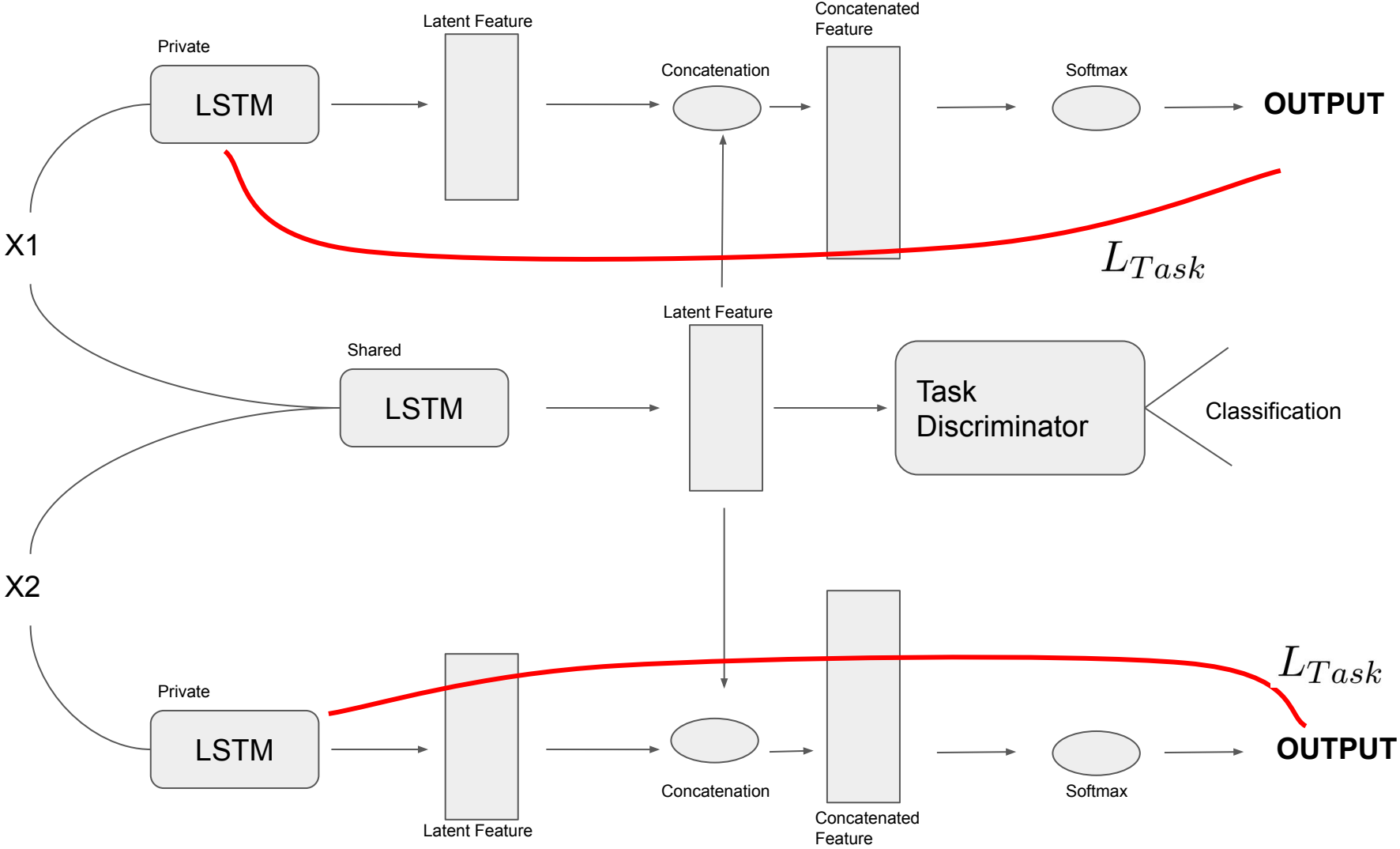
중복 latent representation을 낮춤

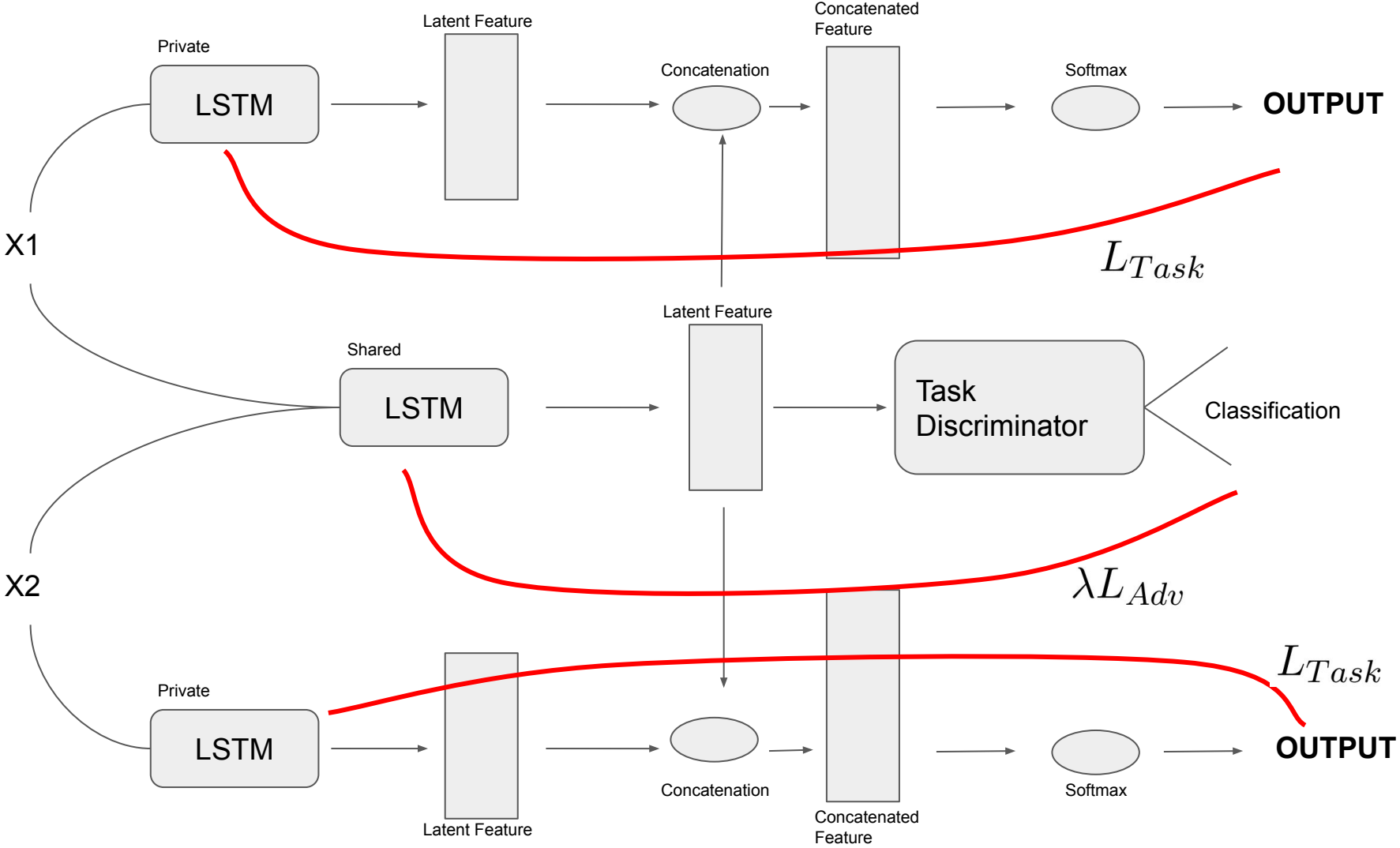
Objective function

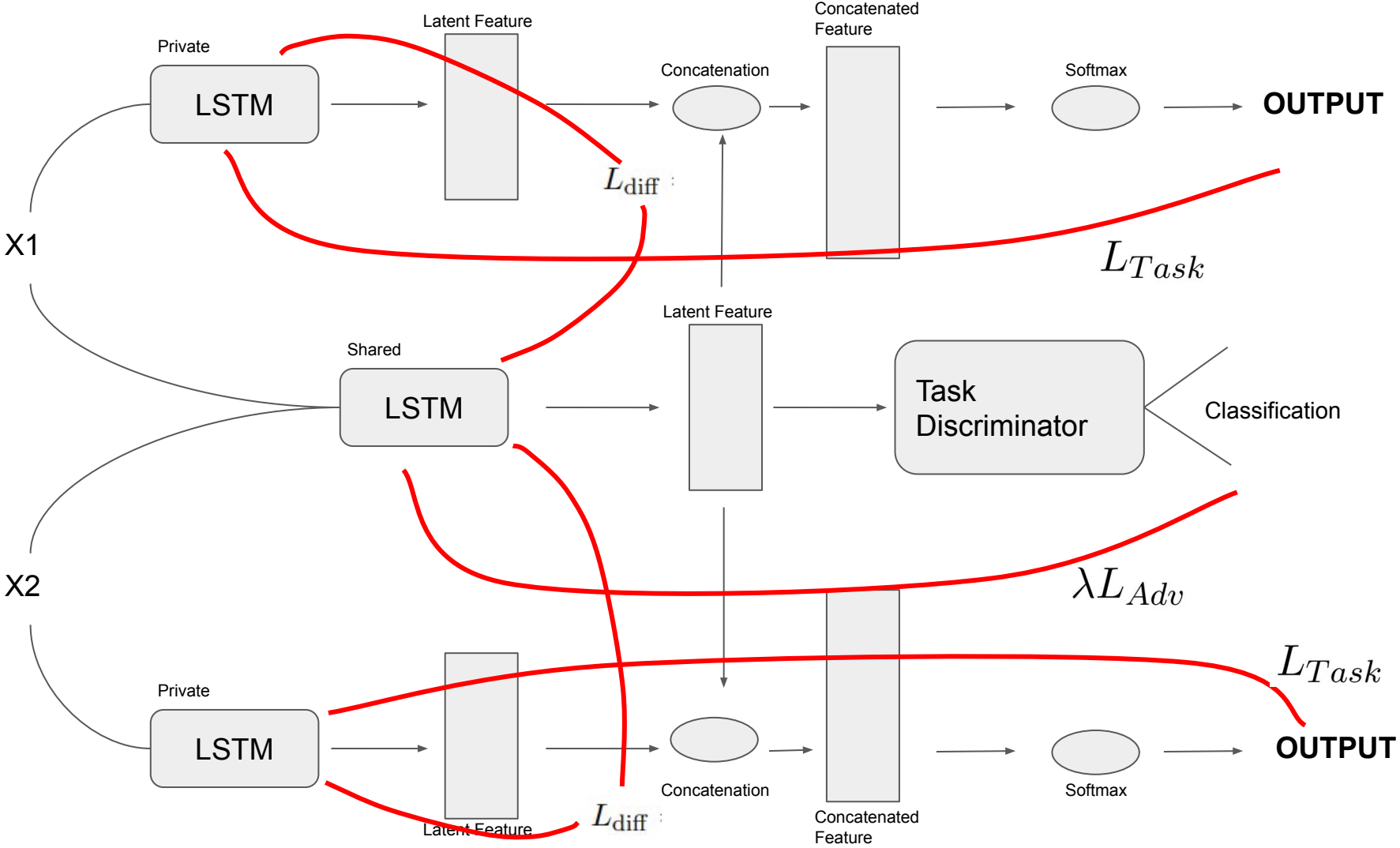
$$L_{\text{diff}} = \sum_{k=1}^{K} \left\| \mathbf{S}^{k\top} \mathbf{H}^{k} \right\|_F^2 ,$$

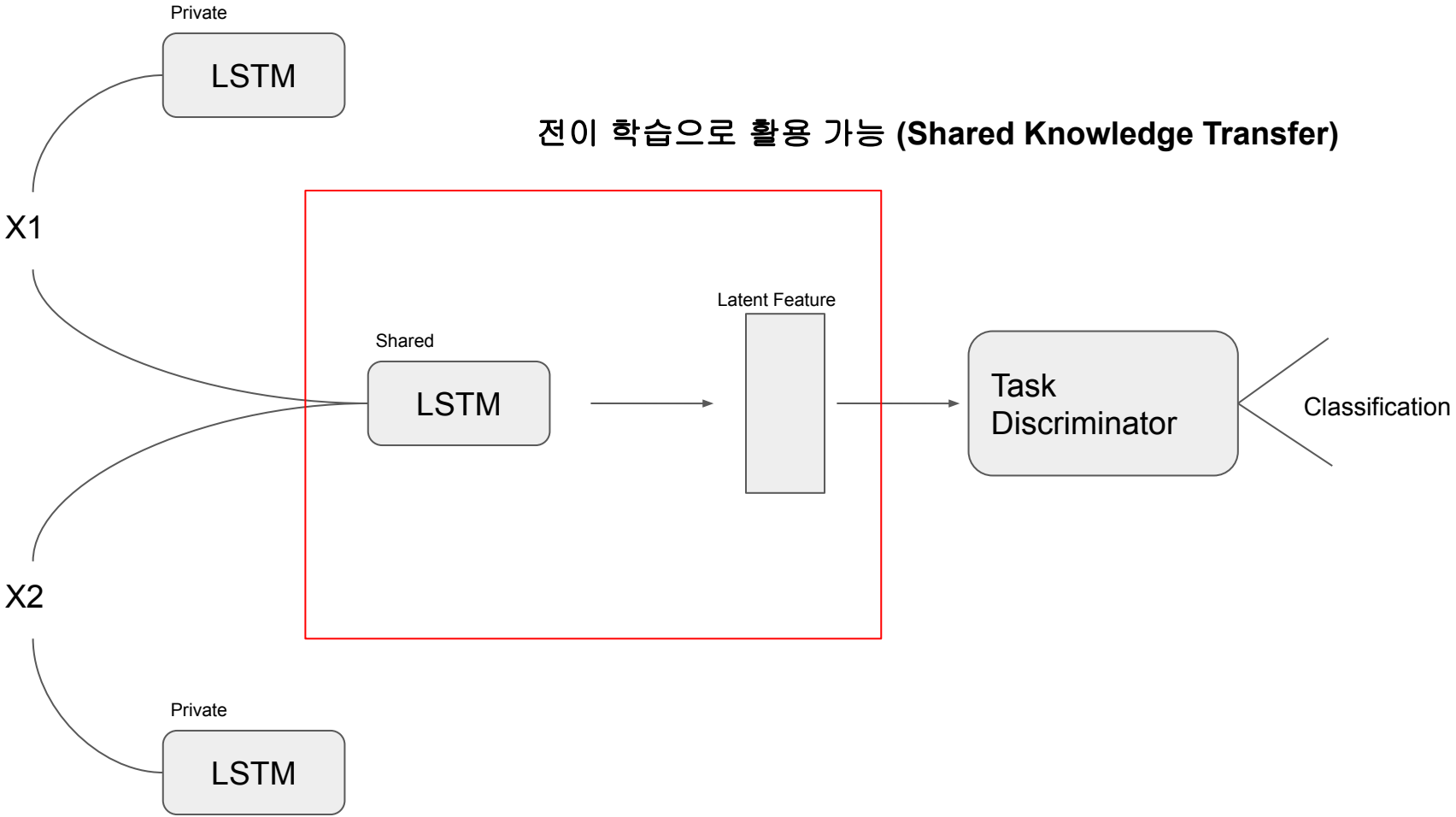
where $\| \cdot \|_F^2$ is the squared Frobenius norm. $\mathbf{S}^k$ and $\mathbf{H}^k$ are two matrics, whose rows are the output of shared extractor $E_s(,;\theta_s)$ and task-specific extrator $E_k(,;\theta_k)$ of a input sentence.


Private
LSTM
Latent Feature
Concatenated Feature
Concatenation
Softmax
OUTPUT
X1
$\mathbf{h}_t^k = \mathbf{LSTM}(x_t, \mathbf{h}_{t-1}^m, \theta_k)$
Latent Feature
Shared
LSTM
Task Discriminator
Classification
$\mathbf{s}_t^k = \mathbf{LSTM}(x_t, \mathbf{s}_{t-1}^k, \theta_s),$
$D(\mathbf{s}_T^k, \theta_D) = \mathrm{softmax}(\mathbf{b} + \mathbf{U}\mathbf{s}_T^k)$
X2
$\mathbf{h}_t^k = \mathbf{LSTM}(x_t, \mathbf{h}_{t-1}^m, \theta_k)$
Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT

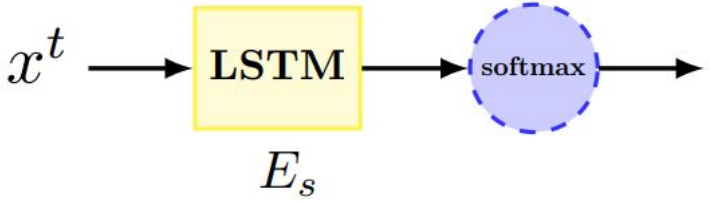

Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT
X1
L_{Task}
Latent Feature
Shared
LSTM
Task Discriminator
Classification
X2
L_{Task}
Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT


Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT
X1
L_{Task}
Latent Feature
Shared
LSTM
Task Discriminator
Classification
X2
λL_{Adv}
L_{Task}
Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT

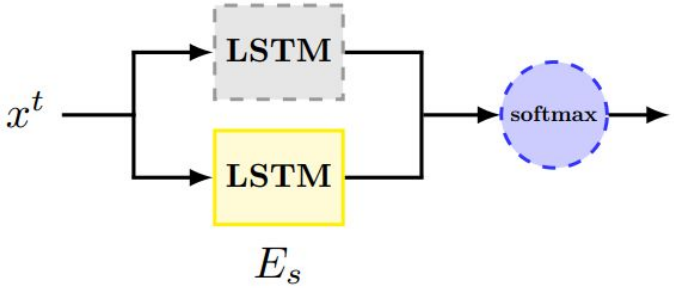

Private
LSTM
Latent Feature
Concatenated Feature
Concatenation
Softmax
OUTPUT
X1
L_{diff}
L_{Task}
Shared
LSTM
Latent Feature
Task Discriminator
Classification
X2
λL_{Adv}
L_{Task}
Private
LSTM
Latent Feature
Concatenation
Concatenated Feature
Softmax
OUTPUT
L_{diff}


Private
LSTM
전이 학습으로 활용 가능 (Shared Knowledge Transfer)
X1
Latent Feature
Shared
LSTM
Task
Discriminator
Classification
X2
Private
LSTM

전이 학습 (Shared Knowledge Transfer)

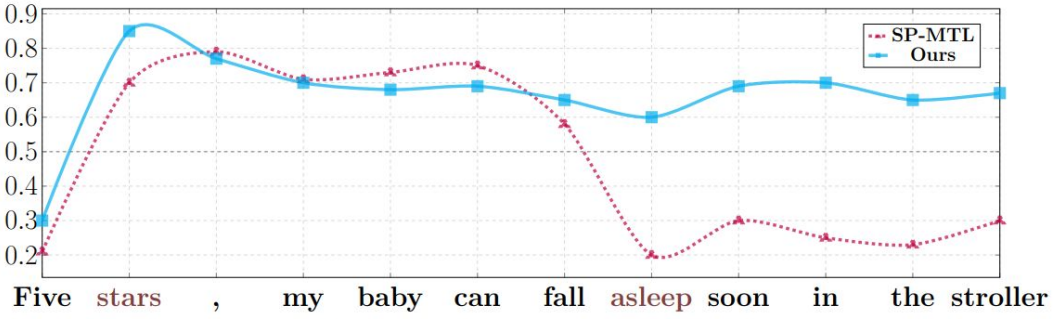


(a) Single Channel



(b) Bi-Channel

Shared FS 와 Private FS의 구분의 중요성



(a) Predicted Sentiment Score by Two Models

“Asleep”이라는 단어에서 갑자기 부정적인 sentiment로 예측

“Asleep”이라는 단어가 shared feature space에 존재 하고, 다른 task에서는 부정적인 의미로 사용되었다고 추측

사용된 개념

- Adversarial Network (from GAN) https://arxiv.org/abs/1406.2661
- Domain Adaptation https://www.alexkulesza.com/pubs/adapt_mlj10.pdf
- Orthogonality Constraints https://mitpress.mit.edu/books/advances-neural-information-processing-systems

실제 코드

- https://github.com/ChenglongChen/tensorflow-ASP-MTL/blob/master/core/model.py
- https://github.com/FrankWork/fudan_mtl_reviews