# Semi-Supervised Sequence Modelling with Cross-View Training (CVT)

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, Quoc V. Le
Stanford University | Google Brain

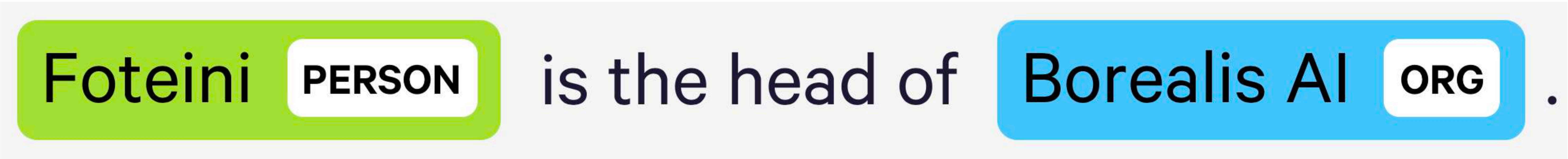
Hareesh Bahuleyan
Borealis AI

# Problem Formulation

- **Labelled data**
  - Set of sequences with human annotated labels (e.g. sentiment)
- **Unlabelled data**
  - No labels (but data from the same domain)
  - Larger set

# Sequence Tagging (NER) as Example

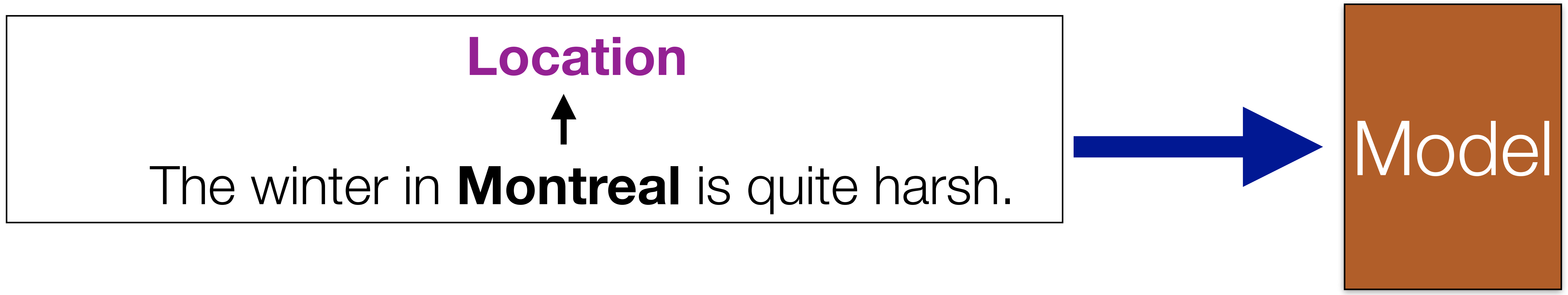
- Predict a tag for each token in the sequence: **Person**, **Location**, **Organization**, **Other**




SpaCy NER: https://explosion.ai/demos/displacy-ent

# Classic Self-Training

1. Train the model with labelled data

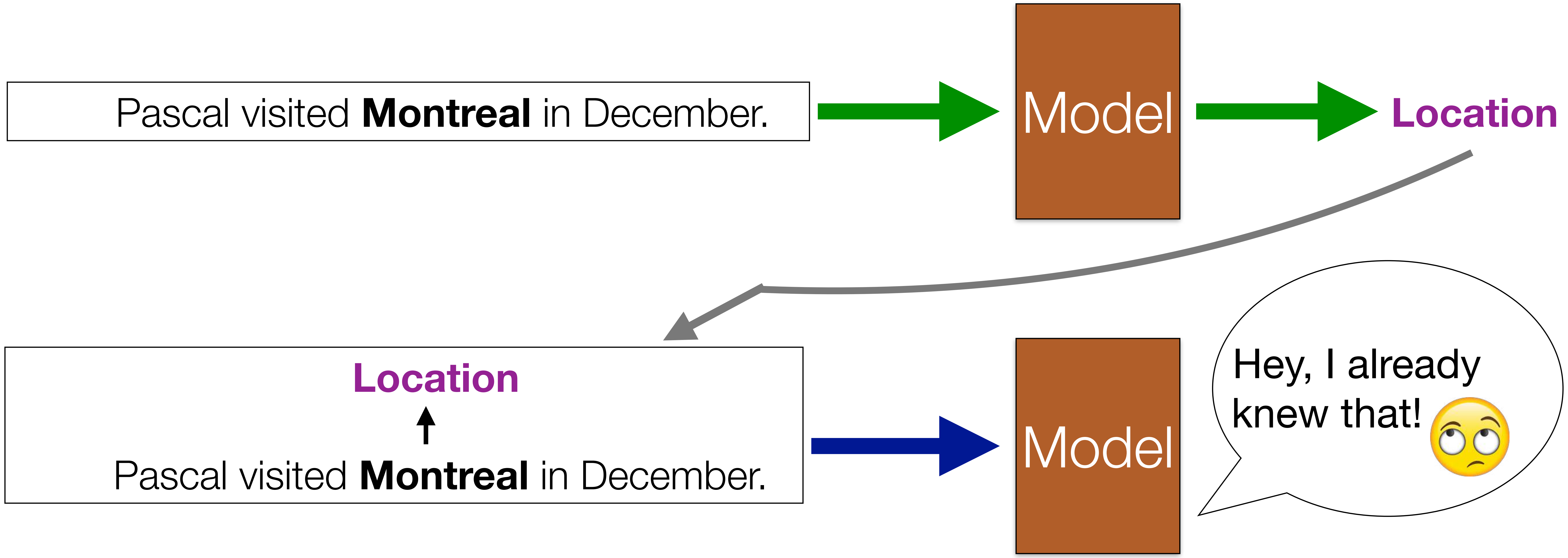



2. Ask the model to provide a prediction on an unlabelled data

- machine-provided pseudo label to additionally train the model

# Classic Self-Training

We are feeding the same information that the model already knows, back to the model !

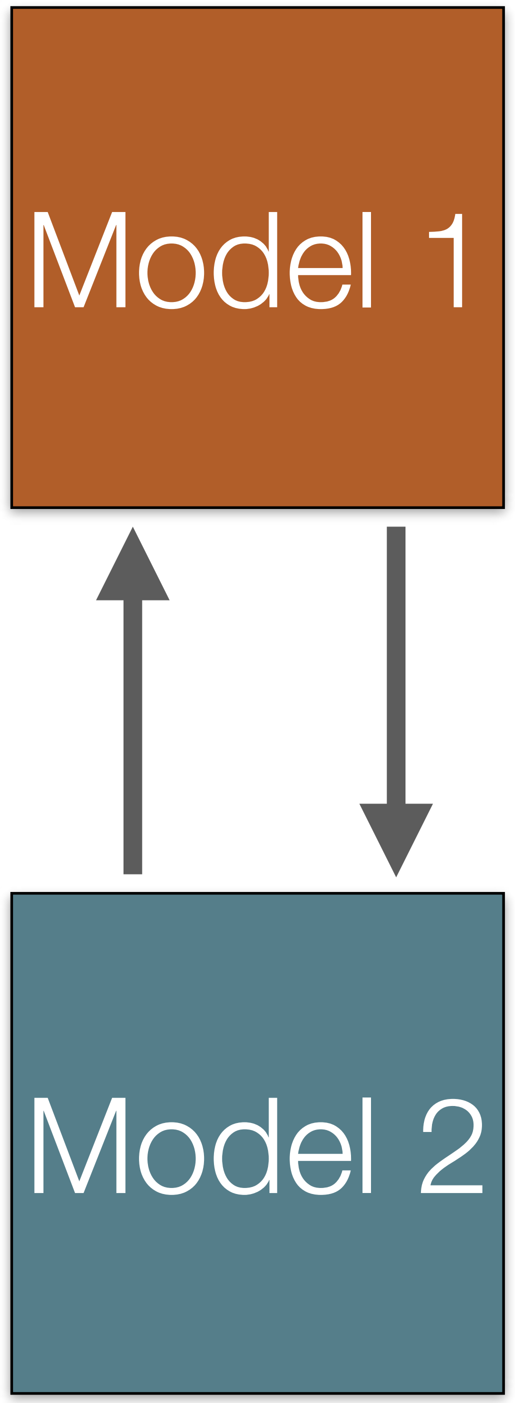



This approach seems tautological or circular !

# Revisiting Co-Training

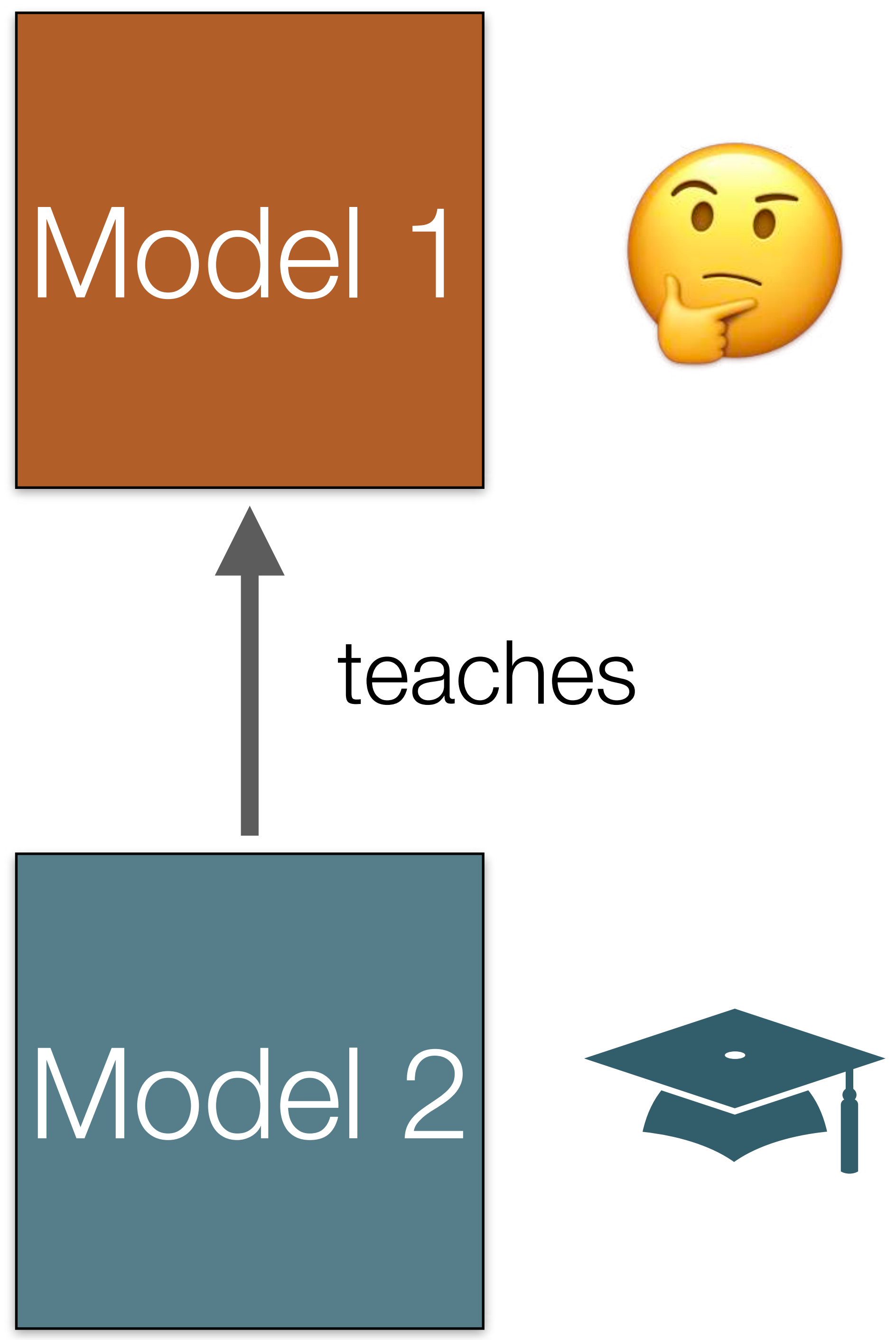
(Blum and Mitchell, 1998)

- In previous setup, the same model acts as both a teacher and a student that is trained on those predictions
- **Co-training :**
  - Two models trained with disjoint views of the input
  - On unlabeled data, each one acts as a "teacher" for the other model



# Disjoint Views Example

- Example: “I wandered around the streets of **Montreal**”
- Model 1 sees “I wandered around ___ ____ ____”
- Model 2 sees “___ ____ ____ the streets of Montreal”



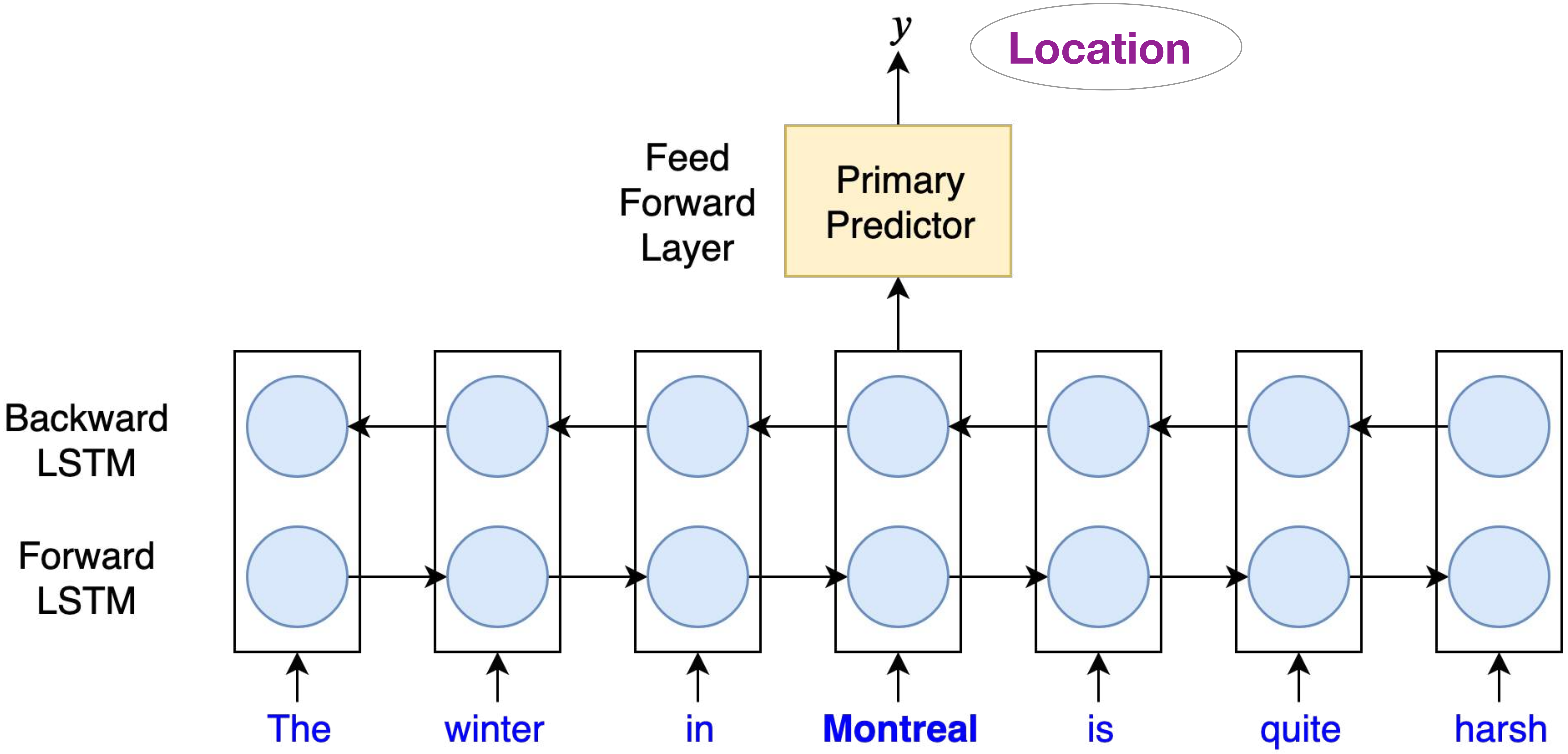
# Adopting Co-Training to Neural Nets

- Two separate models, each one alone is not going to be great by itself - since they see only part of the input.
- A neural network with some layers/parts shared, and other layers/parts are independent.
- **Primary Predictor:** They propose to train an additional model that sees the **whole** input sentence.
    - It can also be used at test time to make a prediction
- **Auxiliary Predictors:** see restricted views of the input
- Trained to produce **consistent predictions across different views of the input**
- The quality of representations are improved and more robust

# Learning on Labelled Example

- Primary Predictor

- Cross-Entropy Loss

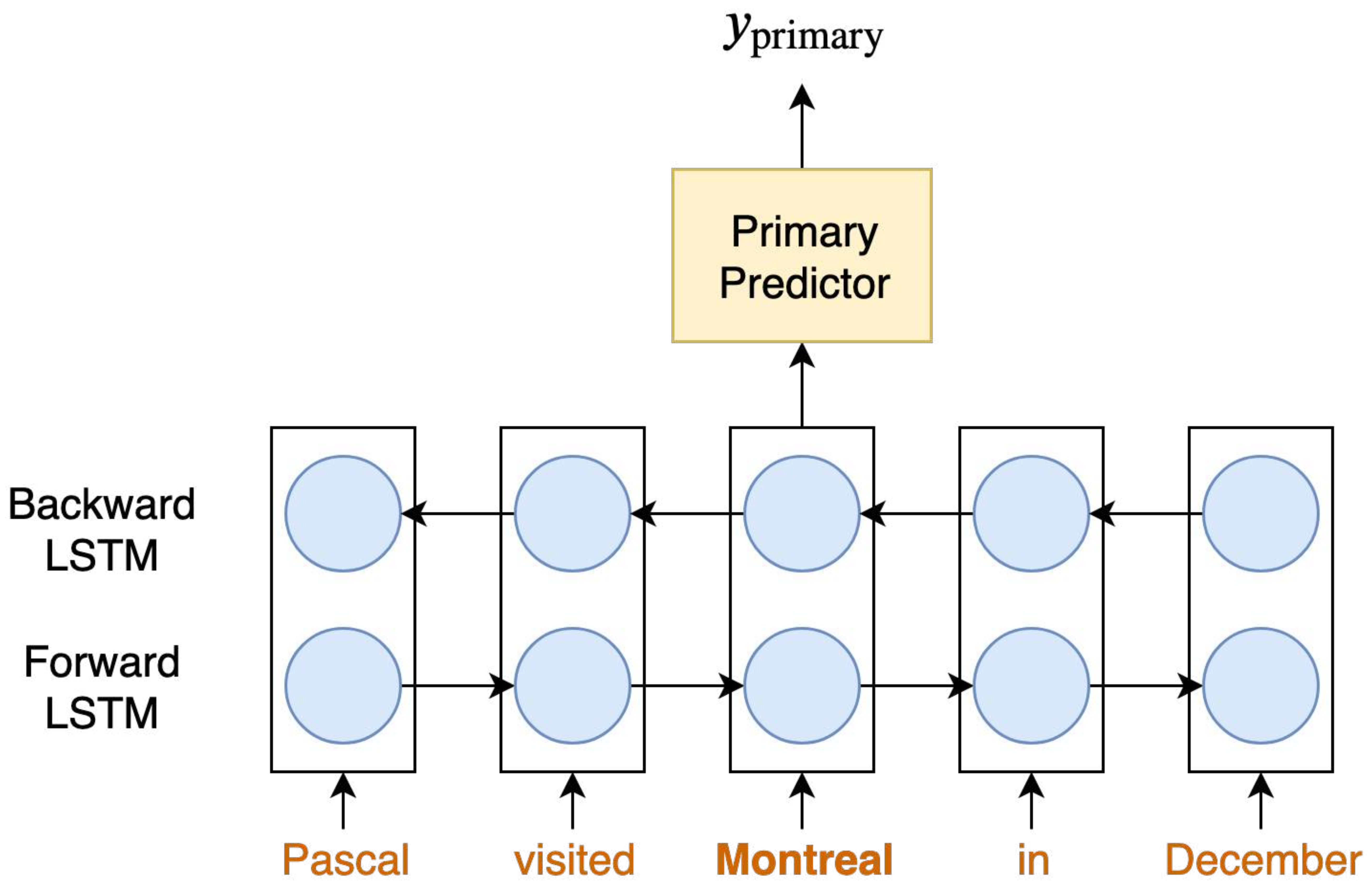



$$\mathcal{L}_{\text{sup}}(\theta) = \frac{1}{|\mathcal{D}_l|} \sum_{x_i, y_i \in \mathcal{D}_l} CE(y_i, p_\theta(y|x_i))$$

# An what about Unlabelled examples?

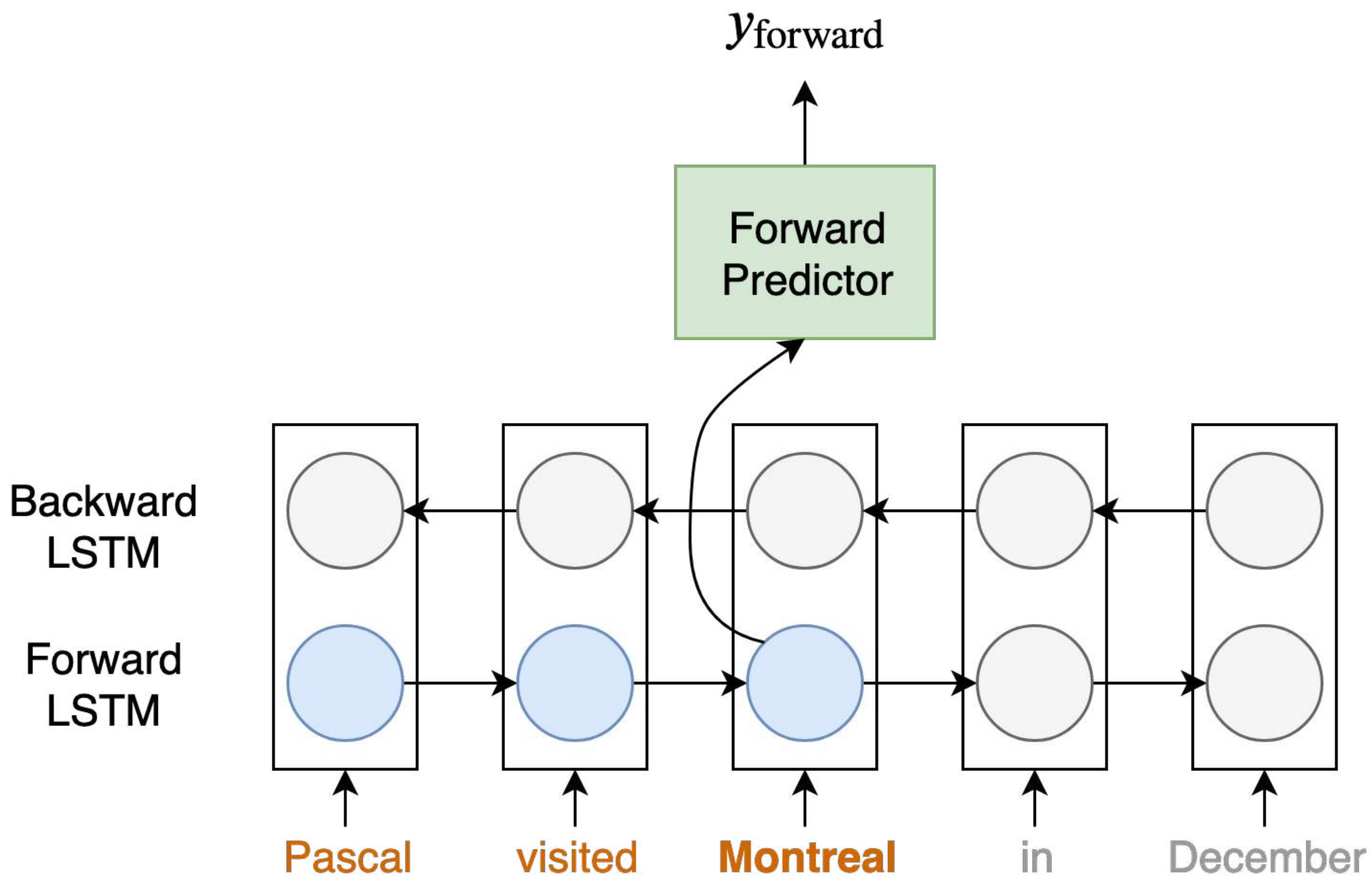
# Primary Predictor

- Do forward pass with the primary predictor has full view.
- Prediction module: feed forward layer followed by softmax



# Auxiliary Predictor 1: Forward Predictor

- Partial view of the sentence
- Intermediate output of Forward LSTM
- Separate prediction module



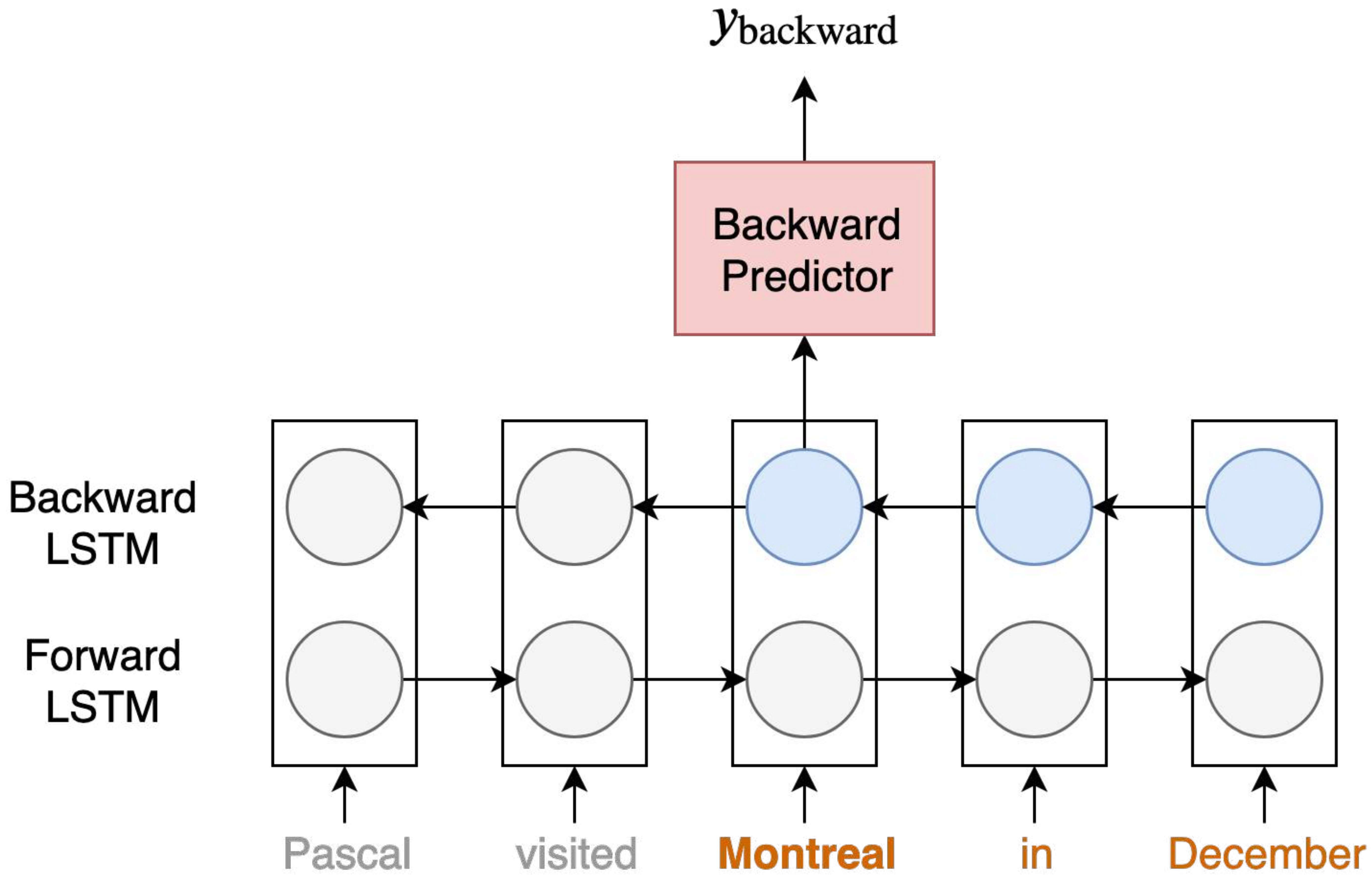
# Unlabelled Example Loss Computation

- On the unlabelled examples, loss function encourages *y_forward* to MATCH *y_primary*
- **Minimizing KL divergence between the probabilities from the primary prediction module and the auxiliary module**

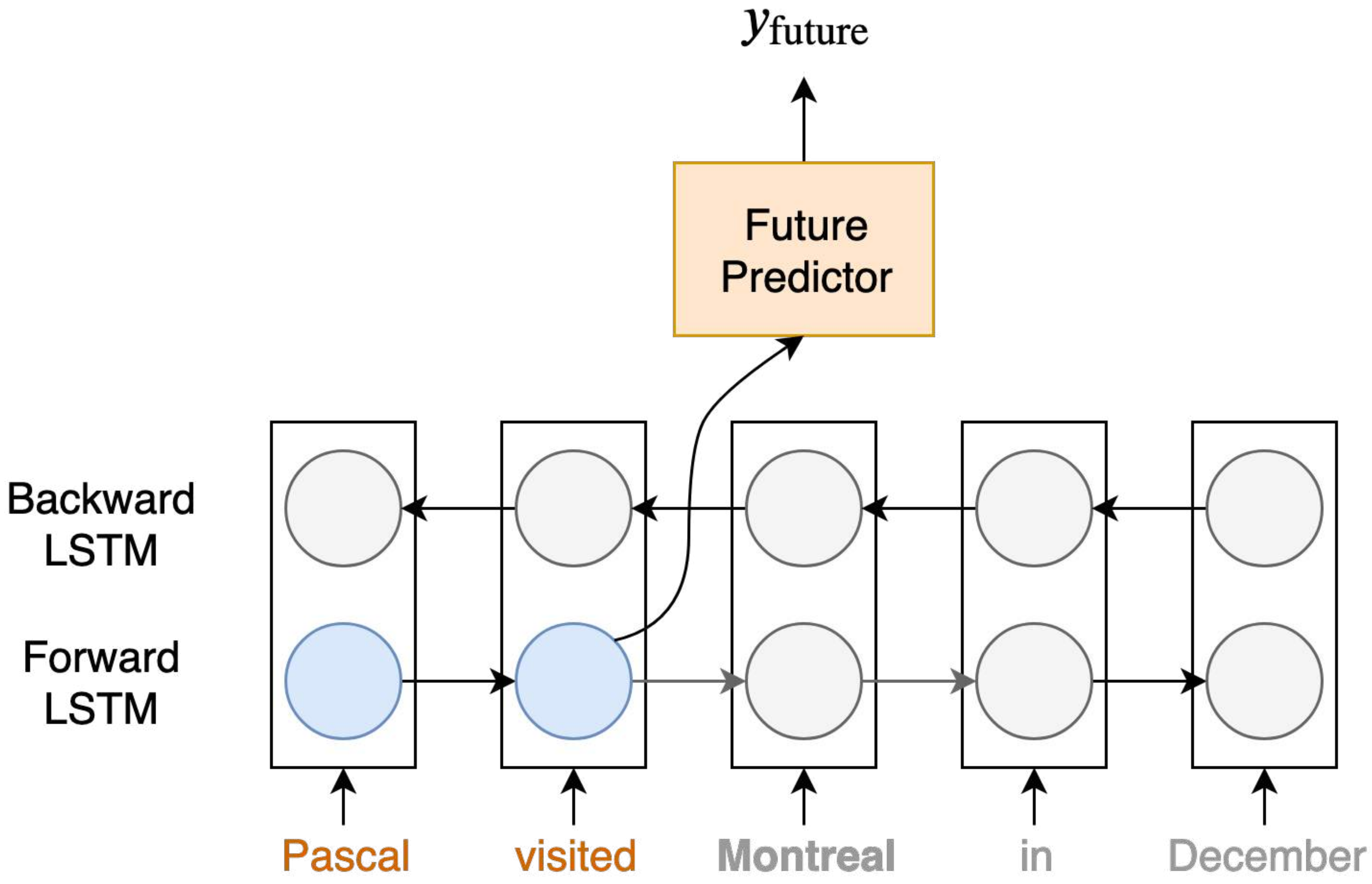
$$\boxed{\mathcal{L}_{\text{CVT}}(\theta) = \frac{1}{|\mathcal{D}_{ul}|} \sum_{x_i \in \mathcal{D}_{ul}} \sum_{j=1}^{k} D(p_\theta(y|x_i), p_\theta^j(y|x_i))}$$

- The hope is that the forward predictor ***learns*** from the primary predictor which sees more information
- Back-prop through the auxiliary module (but not the primary module):
  - improves the representations from the layers/parts that are shared between the auxiliary and the primary
  - which will in turn improve the primary module
- Combine the supervised and CVT losses: $\boxed{\mathcal{L} = \mathcal{L}_{\text{sup}} + \mathcal{L}_{\text{CVT}}}$
  - alternate minimizing $\mathscr{L}_{\text{sup}}$ over a mini-batch of labeled examples and minimizing $\mathscr{L}_{\text{CVT}}$ over a mini-batch of unlabeled examples

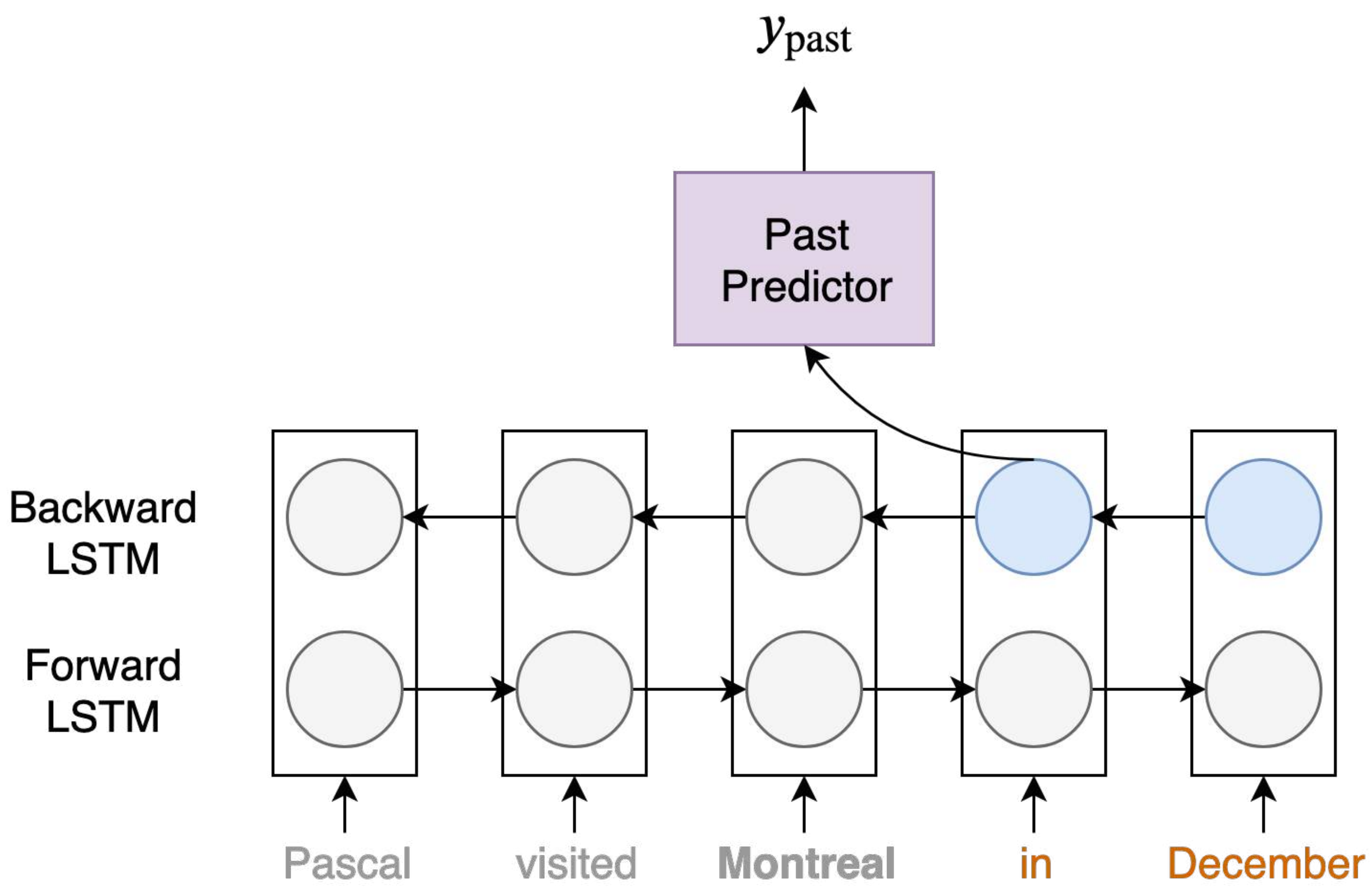
# Auxiliary Predictor 2: Backward Predictor



# Auxiliary Predictor 3: Future Predictor

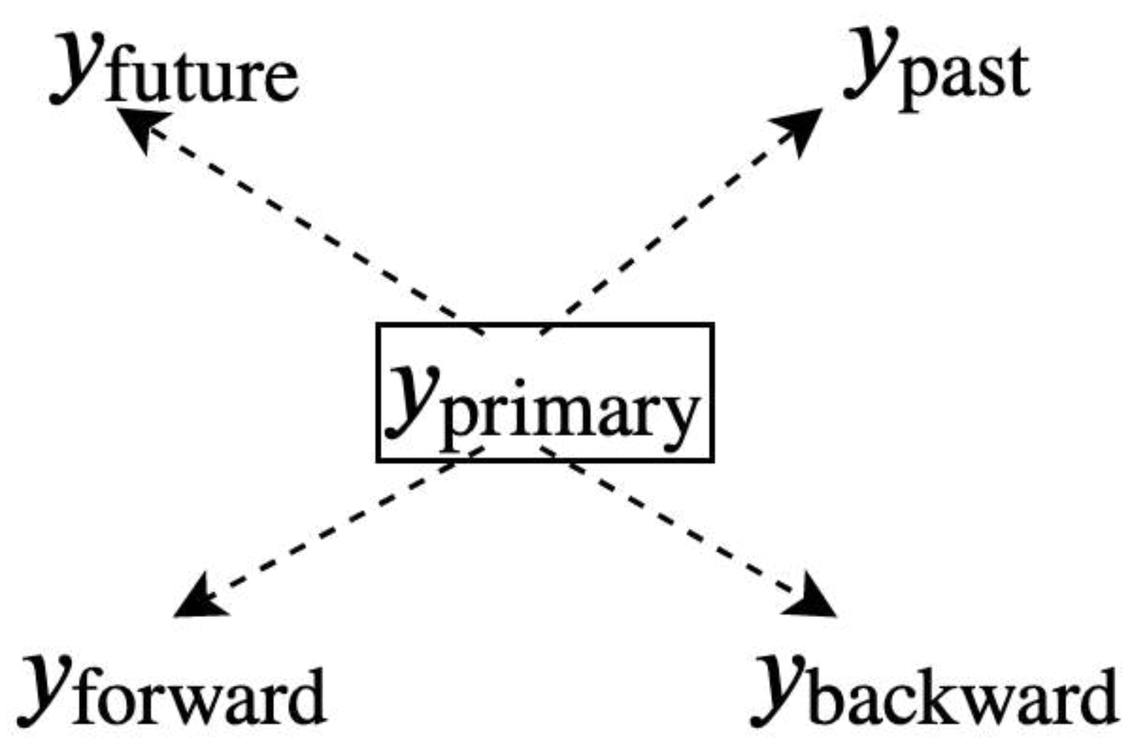


# Auxiliary Predictor 4: Past Predictor



# Putting it all together …

- Labelled example - regular cross entropy
- Cross-View Training on unlabelled example:
    - Consistency between the predictions of the primary and the auxiliary modules
- At test time, only the primary module is needed to make a prediction



# Combining with Multi-task Training

# Multi-task Learning

- Dataset A labelled for task A, Dataset B labelled for Task B and so on…
- Add a set of **more predictors** on top of the Bi-LSTM, one for each task (such as NER, POS tagging, etc.).
- During supervised learning, **randomly select a task** and then update $\mathscr{L}_{\text{sup}}$ using a mini-batch of labeled data for that task.
- On unlabelled examples, <u>jointly train across all tasks at once</u>:
  - first running forward pass with all the primary prediction modules and then;
  - learning from the predictions with all the auxiliary prediction modules.
- Datasets labeled for multiple tasks are useful for multi-task systems to learn from, but most datasets are only labeled with one task!
  - A benefit of multi-task CVT is that the model creates (artificial) <u>all-tasks-labeled examples</u> from unlabeled data.

# Experiments - Tasks

- 7 Tasks:
  - Named Entity Recognition
  - Text Chunking
  - Combinatory Categorial Grammar (CCG) Super-tagging
  - Fine-Grained NER
  - Part-of-Speech (POS) Tagging
  - Dependency Parsing
  - Machine Translation
- 1BillionWord Corpus as a source of unlabelled data

# Experiments - Baselines

- **Word Dropout**
  - Only primary prediction module
  - In student mode: replace words with REMOVED token
- **Virtual Adversarial Training** (Miyato et al., 2016)
  - Only primary prediction module
  - Add noise (chosen adversarially) to the word embeddings of the student
- **ELMo** (Peters et al., 2018)
  - Language model pre-training

# Comparing CVT self-training and LM pre-training

- **CVT**: learn representations targeted to a particular task

✓ Comparable or better accuracy

✓ Works great in combination with multi-task learning

✦ Requires (roughly) in-domain unlabeled data

✦ Have to re-train for each task

- **Pre-training**: learn useful "general" representations

✓ Useful for many tasks

✦ Big models, slow pipelined training procedure

# Results

| Method | CCG Acc. | Chunk F1 | NER F1 | FGN F1 | POS Acc. | Dep. Parse UAS | Dep. Parse LAS | Translate BLEU |
|---|---|---|---|---|---|---|---|---|
| Shortcut LSTM (Wu et al., 2017) | 95.1 | | | | 97.53 | | | |
| ID-CNN-CRF (Strubell et al., 2017) | | | 90.7 | 86.8 | | | | |
| JMT† (Hashimoto et al., 2017) | | 95.8 | | | 97.55 | 94.7 | 92.9 | |
| TagLM* (Peters et al., 2017) | | 96.4 | 91.9 | | | | | |
| ELMo* (Peters et al., 2018) | | | 92.2 | | | | | |
| Biaffine (Dozat and Manning, 2017) | | | | | | 95.7 | 94.1 | |
| Stack Pointer (Ma et al., 2018) | | | | | | 95.9 | 94.2 | |
| Stanford (Luong and Manning, 2015) | | | | | | | | 23.3 |
| Google (Luong et al., 2017) | | | | | | | | 26.1 |
| Supervised | 94.9 | 95.1 | 91.2 | 87.5 | 97.60 | 95.1 | 93.3 | 28.9 |
| Virtual Adversarial Training* | 95.1 | 95.1 | 91.8 | 87.9 | 97.64 | 95.4 | 93.7 | – |
| Word Dropout* | 95.2 | 95.8 | 92.1 | 88.1 | 97.66 | 95.6 | 93.8 | 29.3 |
| ELMo (our implementation)* | 95.8 | 96.5 | 92.2 | 88.5 | 97.72 | 96.2 | 94.4 | 29.3 |
| ELMo + Multi-task*† | 95.9 | 96.8 | 92.3 | 88.4 | **97.79** | 96.4 | 94.8 | – |
| CVT* | 95.7 | 96.6 | 92.3 | 88.7 | 97.70 | 95.9 | 94.1 | **29.6** |
| CVT + Multi-task*† | 96.0 | 96.9 | 92.4 | 88.4 | 97.76 | 96.4 | 94.8 | – |
| CVT + Multi-task + Large*† | **96.1** | **97.0** | **92.6** | **88.8** | 97.74 | **96.6** | **95.0** | – |

# An Interesting Case

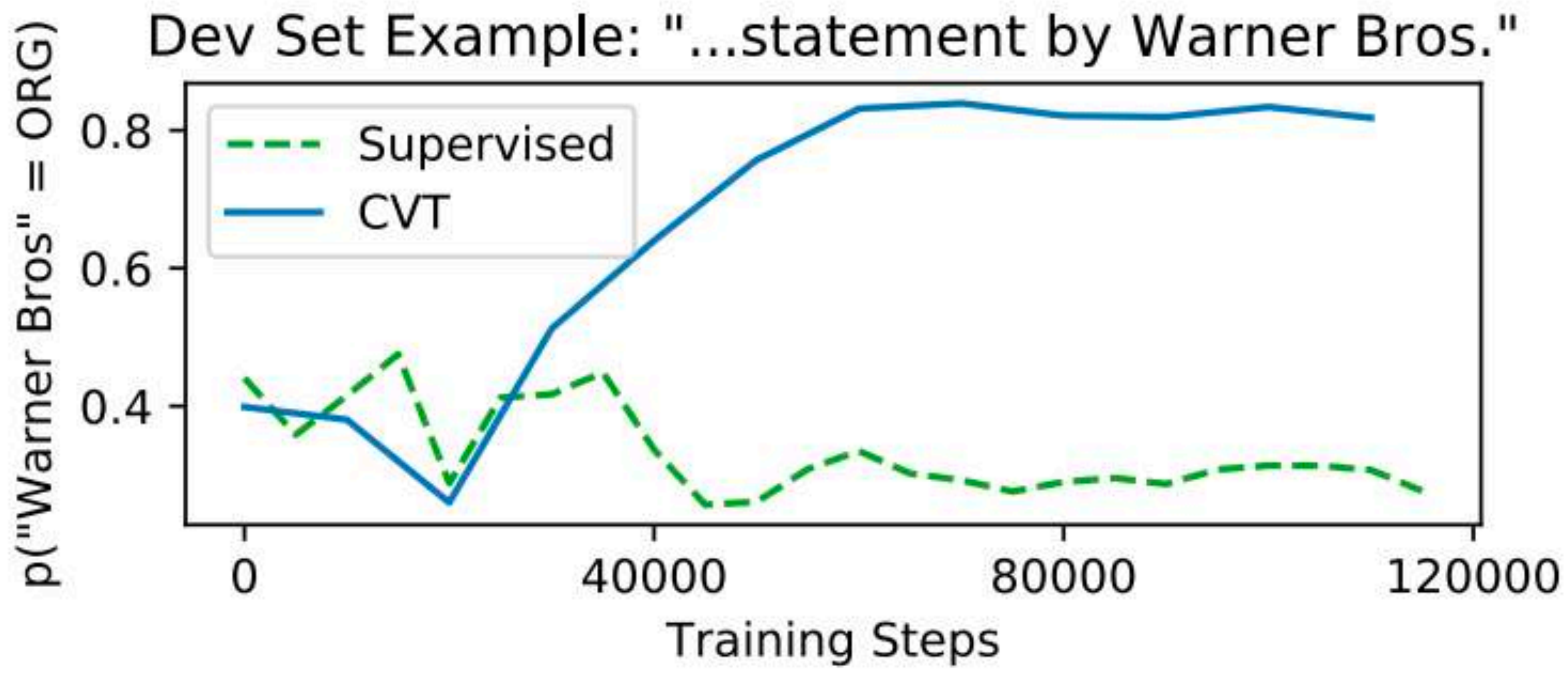
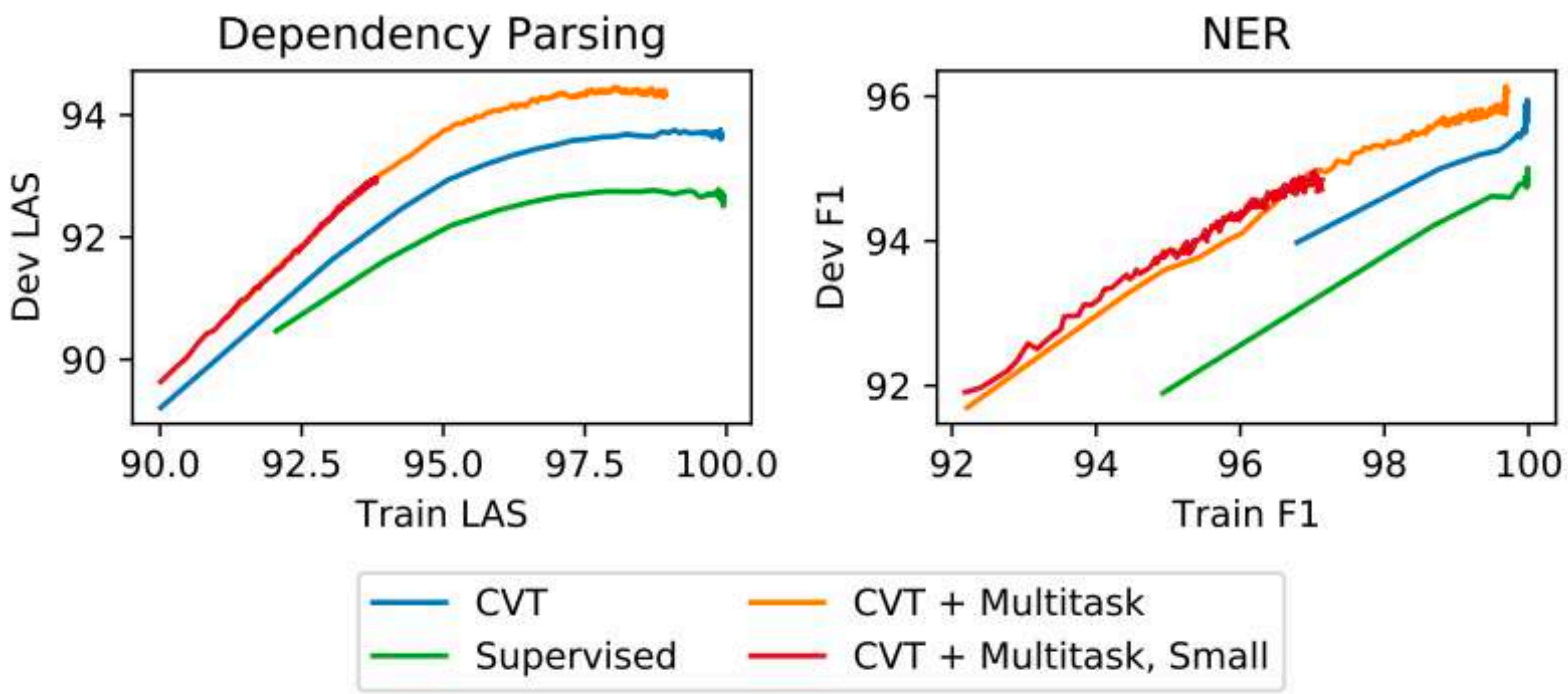


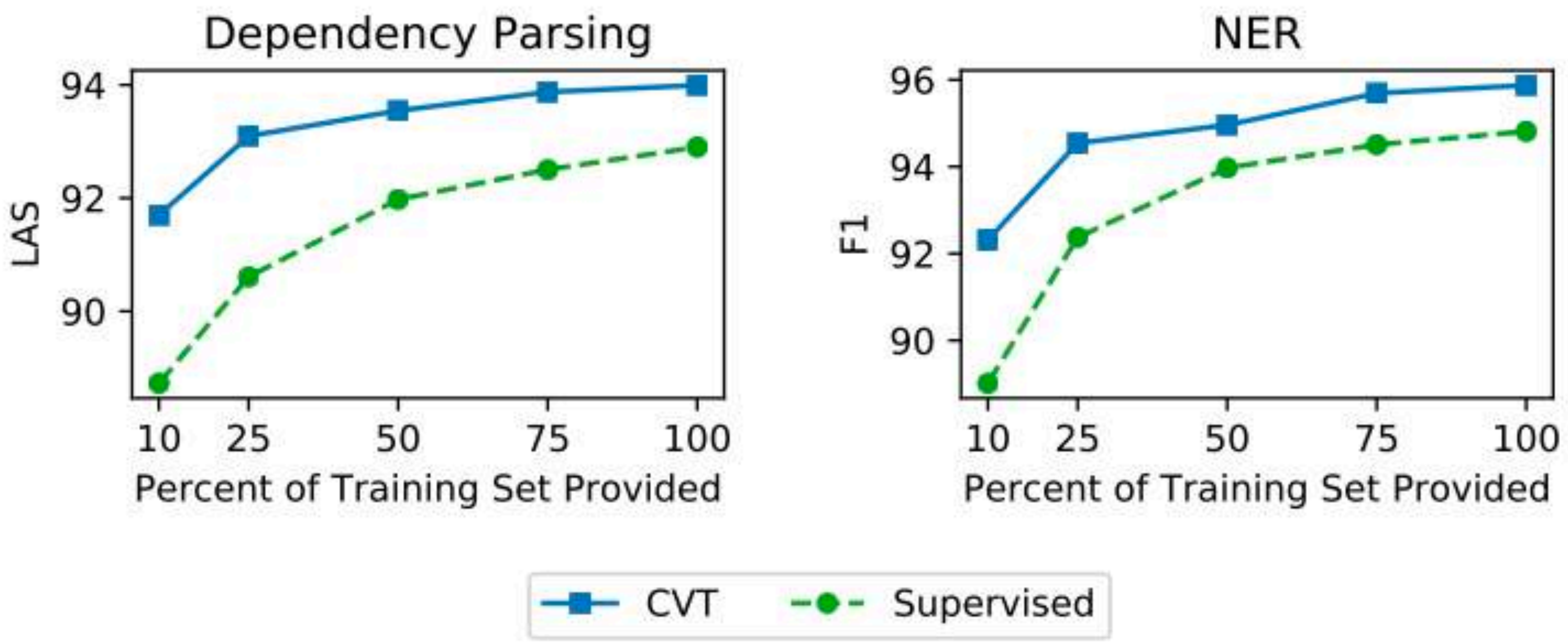

Figure 3: An NER example that CVT classifies correctly but supervised learning does not. "Warner" only occurs as a last name in the train set, so the supervised model classifies "Warner Bros" as a person. The CVT model also mistakenly classifies "Warner Bros" as a person to start with, but as it sees more of the unlabeled data (in which "Warner" occurs thousands of times) it learns that "Warner Bros" is an organization.

# Effectiveness of combining CVT with Multi-Task training



# Performance with size of Labelled Set

- Using only 25% of the labeled data, CVT performs as well or better than a fully supervised model using 100% of the training data!
- Demonstrates that CVT is particularly useful on low resource setting



# Does CVT provide generalizable representations?

- Training the CVT+multi-task model on five tasks
- Freeze the encoder, and then only training a prediction module on the sixth task (fine-tuning).
- This tests whether the encoder's representations generalize to a new task not seen during its training.

| Model | CCG | Chnk | NER | FGN | POS | Dep. |
|---|---|---|---|---|---|---|
| Supervised | 94.8 | 95.6 | 95.0 | 86.0 | 97.59 | 92.9 |
| CVT-MT frozen | 95.1 | 96.6 | 94.6 | 83.2 | 97.66 | 92.5 |
| ELMo frozen | 94.3 | 92.2 | 91.3 | 80.6 | 97.50 | 89.4 |

Table 4: Comparison of single-task models on the dev sets. "CVT-MT frozen" means we pretrain a CVT + multi-task model on five tasks, and then train only the prediction module for the sixth. "ELMo frozen" means we train prediction modules (but no LSTMs) on top of ELMo embeddings.

# Summary

- CVT: A method that uses a mix of labeled and unlabeled data
- On labeled examples —> standard supervised learning
- On unlabeled examples —> CVT teaches auxiliary prediction modules that see restricted views of the input (e.g., only part of a sentence) to match the predictions of the full model seeing the whole input

- Results: CVT is particularly effective when combined with multi-task learning
- Five sequence tagging tasks, machine translation, dependency parsing -> achieves state-of-the-art results
- A general framework for semi-supervised learning that can be applied to many tasks

# Thank You

# Machine Translation

- For the seq2seq (machine translation) case, there are two auxiliary predictors
  - For the first one, restricted view of the input is obtained by applying attention dropout, randomly zeroing out a fraction of its attention weights
  - The second one is trained to predict the next word in the target sequence rather than the current one
  - Since there is no target sequence for unlabeled examples, <u>cannot apply teacher forcing</u> to get an output distribution over the vocabulary from the primary decoder at each time step
  - Instead, produce hard targets for the auxiliary modules by running the primary decoder with beam search on the input sequence —> used to train the auxiliary modules