Active Imitation Learning with Noisy Guidance

Kianté Brantley
University of Maryland
kdbrant@cs.umd.edu

Amr Sharaf
University of Maryland
amr@cs.umd.edu

Hal Daumé III
University of Maryland
Microsoft Research
me@hal3.name

Introduction

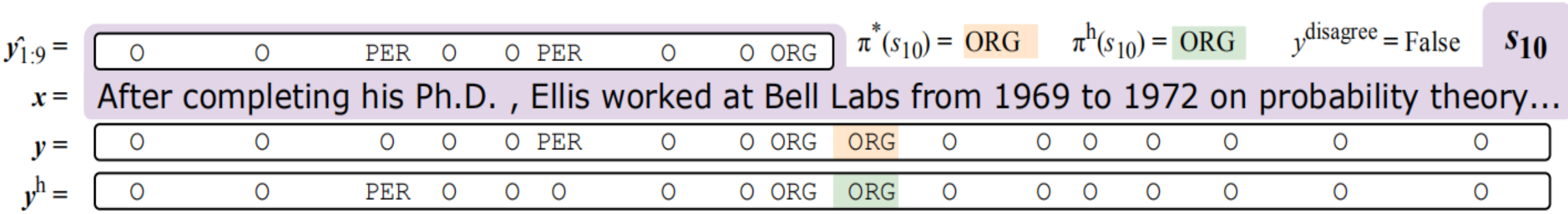
■ Structured Prediction

Instead of predicting a single output, learn models to map inputs to complex **outputs with internal dependencies**, typically requiring a substantial amount of expert-labeled data.

■ Learning to Search

Cast structured prediction as a sequence of smaller classification problems.

As a (degenerate) **imitation learning** task (**Dagger**[1])



[1]A reduction of imitation learning and structured prediction to no-regret online learning. In *AI-Stats*.

DAgger

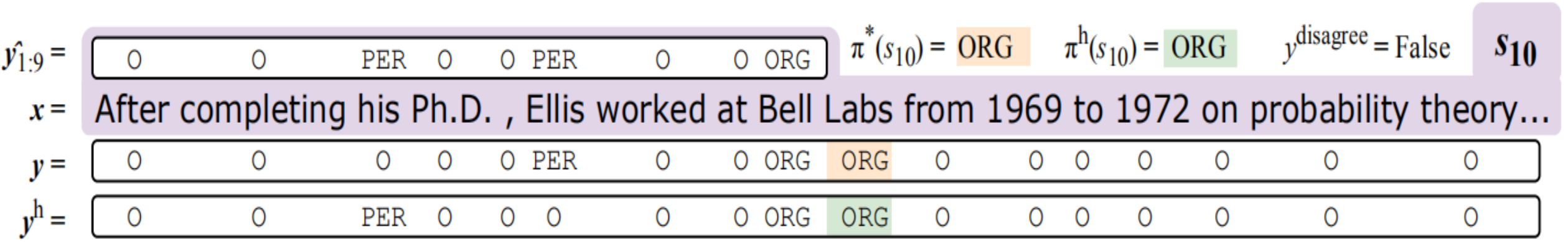
Algorithm 1 DAgger($\Pi, N, \langle\beta_i\rangle_{i=0}^{N}, \pi^\star$)

initialize dataset $D = \{\}$
initialize policy $\hat{\pi}_1$ to any policy in Π
for $i = 1 \ldots N$ **do**
  ▷ *stochastic mixture policy*
  Let $\pi_i = \beta_i \pi^\star + (1 - \beta_i)\hat{\pi}_i$
  Generate a T-step trajectory using π_i
  Accumulate data $D \leftarrow D \cup \{(s, \pi^\star(s))\}$ for all s in those trajectories
  Train classifier $\hat{\pi}_{i+1} \in \Pi$ on D
end for
return best (or random) $\hat{\pi}_i$

Introduction

■ Learning to Query for Imitation

1. LEAQI only asks the expert for a label when it is **uncertain**.
2. LEAQI assumes access to **a noisy heuristic labeling function** (for instance, a rule-based model, dictionary, or inexpert annotator) that can provide low-quality labels
3. Only querying the expert when it thinks the expert is likely to disagree with this label.



Introduction

■ Learning to Query for Imitation

1. LEAQI only asks the expert for a label when it is **uncertain**.
2. LEAQI assumes access to **a noisy heuristic labeling function** (for instance, a rule-based model, dictionary, or inexpert annotator) that can provide low-quality labels
3. Only querying the expert when it thinks the expert is likely to disagree with this label.

- A base model
- A difference classifier

  predicts disagreements between the expert and the heuristic

Method

- **Measuring Policy Certainty** (margin-based)

$$\text{certainty}(\pi, s) = \max_{a} \pi(s, a) - \max_{a' \neq a} \pi(s, a')$$

- **Sampling Probability**[1]

$$Z_i \sim \text{Bern}\left(\frac{b}{b + \text{certainty}(\pi_i, s)}\right)$$

- **Difference Classifier**

$$s_t = [\boldsymbol{w}_t; \text{onehot}(a_{t-1})]$$

$$\hat{d}_i = h_i(s)$$

[1]Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. 2006. Worst-case analysis of selective sampling for linear classification. *JMLR*.

Method

■ Difference Classifier

The challenge in learning the difference classifier is that it must learn based on one-sided feedback

■ Apple Tasting framework[1]

The goal is to avoid sampling too many bad apples and to avoid missing too many good apples.

Minimize Type II errors （以假为真） (it should only very rarely predict “agree” when the truth is “disagree”).

- Increasing sample complexity but **not harming accuracy**

[1]David P. Helmbold, Nicholas Littlestone, and Philip M.Long. 2000. Apple tasting. *Information and Computation*

Method

■ Apple Tasting framework

Random sampling from apples that are predicted to not be tasted and tasting them anyway

$$\sqrt{(m+1)/t}$$

where m is the number of mistakes

t is the number of apples tasted so far

S is the difference dataset

Algorithm 3 AppleTaste_STAP$(S, a_i^{\mathrm{h}}, \hat{d}_i)$

▷ *count examples that are action* a_i^h
let $t = \sum_{(\_,a,\_,\_)\in S} \mathbb{1}[a_i^{\mathrm{h}} = a]$
▷ *count mistakes made on action* a_i^h
let $m = \sum_{(\_,a,\hat{d},d)\in S} \mathbb{1}[\hat{d} \neq d \wedge a_i^{\mathrm{h}} = a]$
$w = \frac{t}{|S|}$ ▷ *percentage of time* a_i^h *was seen*
if $w < 1$ **then**
  ▷ *skew distribution*
  draw $r \sim Beta(1-w, 1)$
else
  draw $r \sim Unifomm(0, 1)$
end if
return $(d = 1) \wedge (r \leq \sqrt{(m+1)/t})$

Algorithm

Algorithm 2 LEAQI$(\Pi, \mathcal{H}, N, \pi^\star, \pi^{\text{h}}, b)$

initialize dataset $D = \{\}$
initialize policy π_1 to any policy in Π
initialize difference dataset $S = \{\}$
initialize difference classifier $h_1(s) = 1\ (\forall s)$
for $i = 1 \ldots N$ **do**
  Receive input sentence $\boldsymbol{x}$
  ▷ *generate a T-step trajectory using π_i*
  Generate output $\hat{\boldsymbol{y}}$ using π_i
  for each s in $\hat{\boldsymbol{y}}$ **do**
    ▷ *draw bernouilli random variable*
    $Z_i \sim \text{Bern}\left(\frac{b}{b+\text{certainty}(\pi_i, s)}\right)$; see §3.3
    if $Z_i = 1$ **then**
      ▷ *set difference classifier prediction*
      $\hat{d}_i = h_i(s)$
      if AppleTaste$(s, \pi^{\text{h}}(s), \hat{d}_i)$ **then**
        ▷ *predict agree query heuristic*
        $D \leftarrow D \cup \{\ (s, \pi^{\text{h}}(s))\ \}$
      else
        ▷ *predict disagree query expert*
        $D \leftarrow D \cup \{\ (s, \pi^\star(s))\ \}$
        $d_i = \mathbb{1}\left[\pi^\star(s) = \pi^{\text{h}}(s)\right]$
        $S \leftarrow S \cup \{\ (s, \pi^{\text{h}}(s), \hat{d}_i, d_i)\ \}$
      end if
    end if
  end for
  Train policy $\pi_{i+1} \in \Pi$ on D
  Train difference classifier $h_{i+1} \in \mathcal{H}$ on S to minimize Type II errors (see §3.2)
end for
return best (or random) π_i

Experiment

1. Does uncertainty-based active learning achieve lower query complexity than passive learning in the learning to search settings?
2. Does learning a difference classifier improve query efficiency over active learning **alone**?
3. Does Apple Tasting successfully handle the problem of learning from one-sided feedback?
4. Is the approach robust to cases where the noisy heuristic is uncorrelated with the expert?
5. Is casting the heuristic as a policy more effective than using its output as features?

Experiment

■ Baselines（online active learning a single pass over dataset）

1. **DAGGER** Passive Dagger Q1
2. **ACTIVE DAGGER** uncertain Q2
3. **DAGGER+FEAT** Q5

   the heuristic policy`s output appended as an input feature
4. **ACTIVEDAGGER+FEAT** Q5

● Ablations

1. **LEAQI+NOAT** no apple tasting Q3
2. **LEAQI+NOISYHEUR** Q4

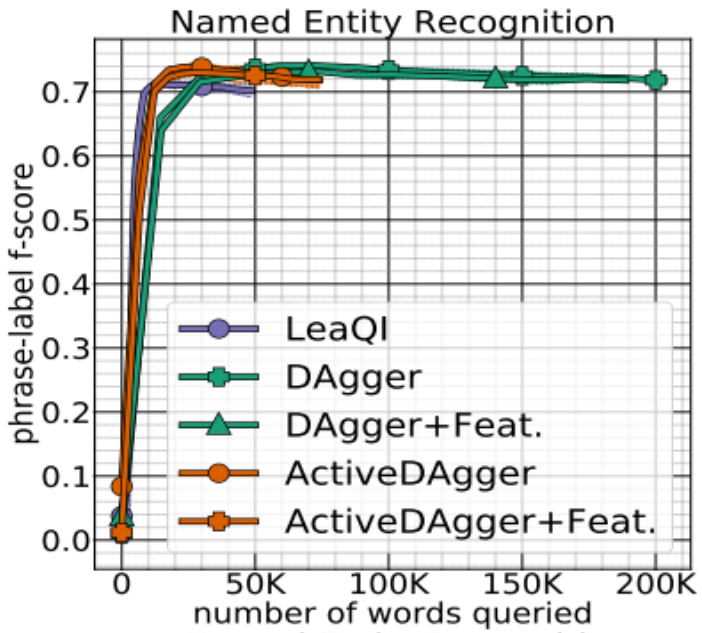
   the heuristic returns a label uniformly at random

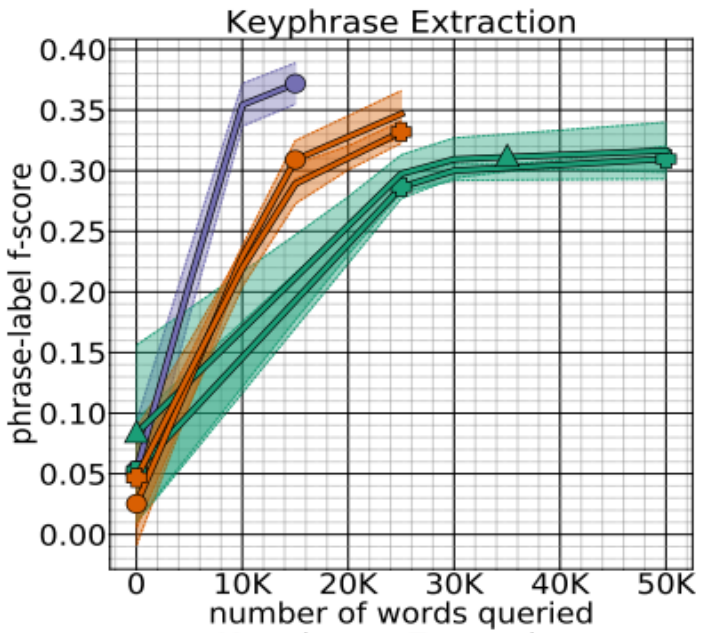
Experiment

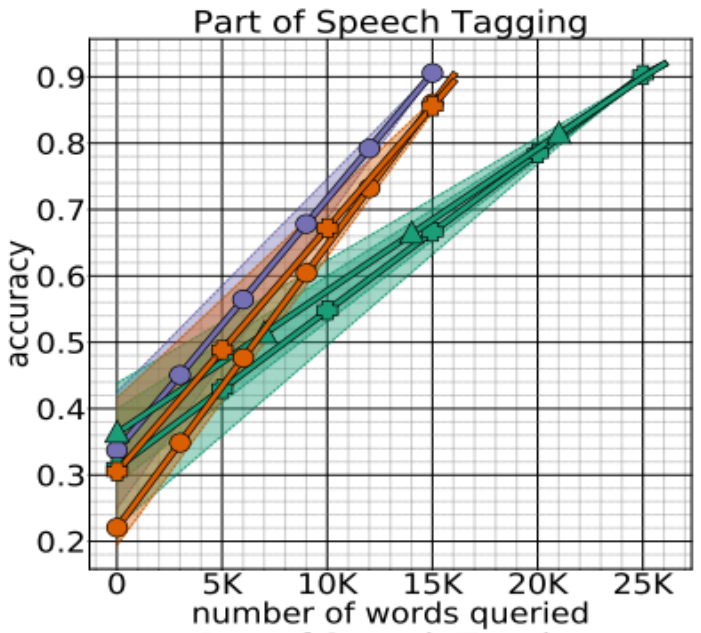
$$s_t = [\boldsymbol{w}_t; \text{onehot}(a_{t-1})]$$

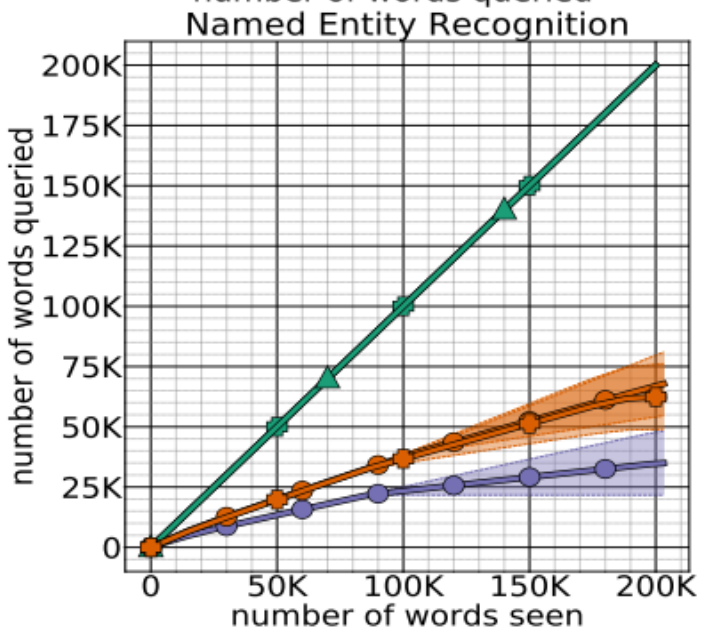
Task	Named Entity Recognition	Keyphrase Extraction	Part of Speech Tagging
Language	English (en)	English (en)	Modern Greek (el)
Dataset	CoNLL'03 (Tjong Kim Sang and De Meulder, 2003)	SemEval 2017 Task 10 (Augenstein et al., 2017)	Universal Dependencies (Nivre, 2018)
# Ex	14,987	2,809	1,662
Avg. Len	14.5	26.3	25.5
# Actions	5	2	17
Metric	Entity F-score	Keyphrase F-score	Per-tag accuracy
Features	English BERT (Devlin et al., 2019)	SciBERT (Beltagy et al., 2019)	M-BERT (Devlin et al., 2019)
Heuristic	String matching against an offline gazeteer of entities from Khashabi et al. (2018)	Output from an unsupervised keyphrase extraction model Florescu and Caragea (2017)	Dictionary from Wiktionary, similar to Zesch et al. (2008) and Haghighi and Klein (2006)
Heur Quality	P 88%, R 27%, F 41%	P 20%, R 44%, F 27%	10% coverage, 67% acc

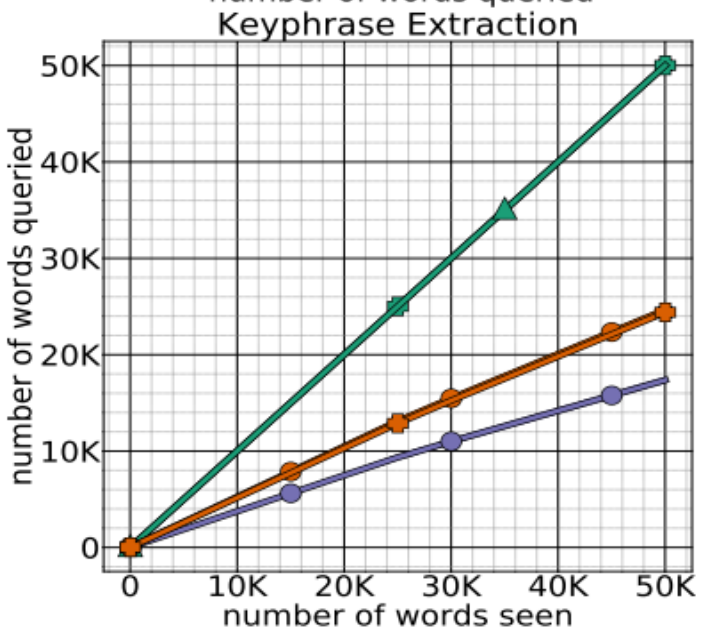
Experiment

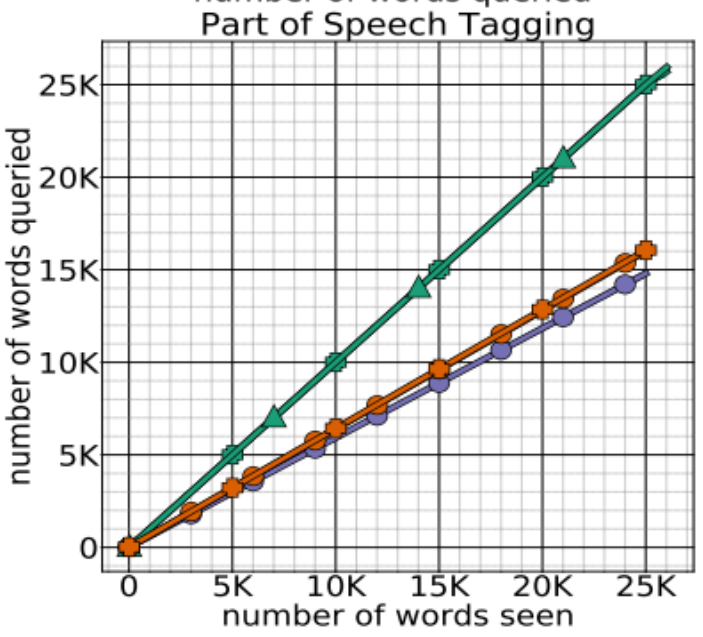

Named Entity Recognition
phrase-label f-score
number of words queried
LeaQI
DAgger
DAgger+Feat.
ActiveDAgger
ActiveDAgger+Feat.


Keyphrase Extraction
phrase-label f-score
number of words queried


Part of Speech Tagging
accuracy
number of words queried


Named Entity Recognition
number of words queried
number of words seen


Keyphrase Extraction
number of words queried
number of words seen


Part of Speech Tagging
number of words queried
number of words seen

Experiment

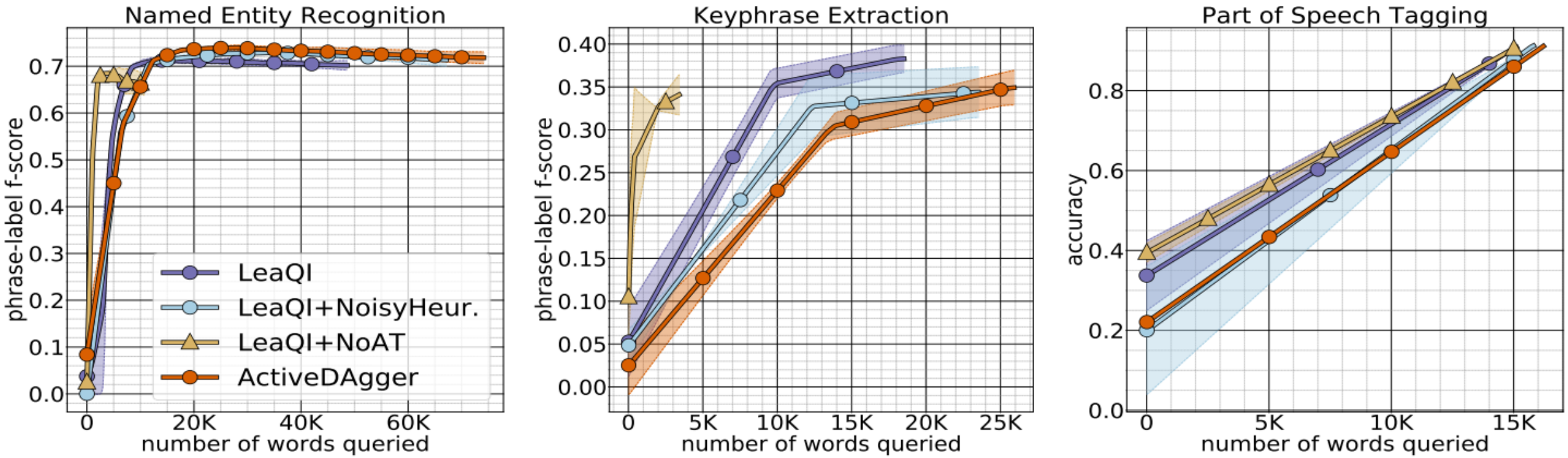
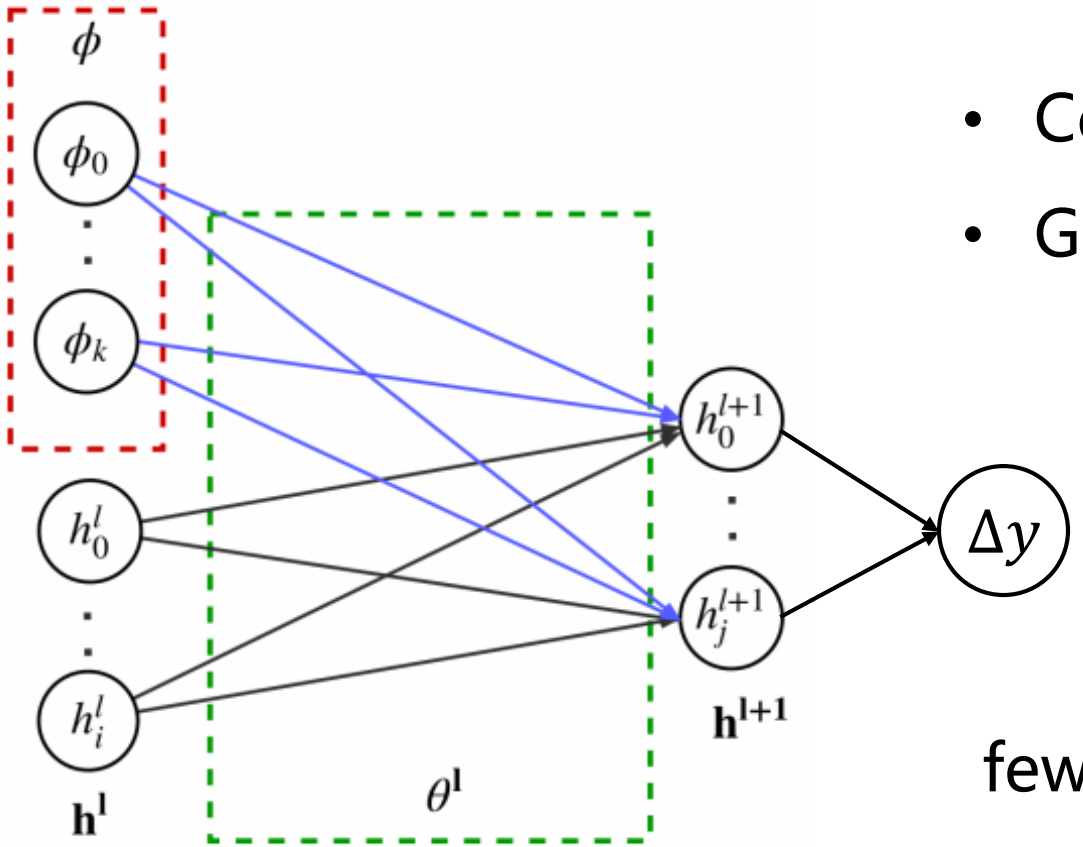


Figure 3: Ablation results on (left) named entity recognition, (middle) keyphrase extraction and (right) part of speech tagging. In addition to LEAQI and DAgger (copied from Figure 2), these graphs also show LEAQI+NOAT (apple tasting disabled), and LEAQI+NOISYHEUR. (a heuristic that produces labels uniformly at random).

改进

x

$\phi(\mathrm{x}, \mathrm{L}, U, f)$



- Context parameters ϕ
- Global parameters θ

few shot learning ϕ on New task