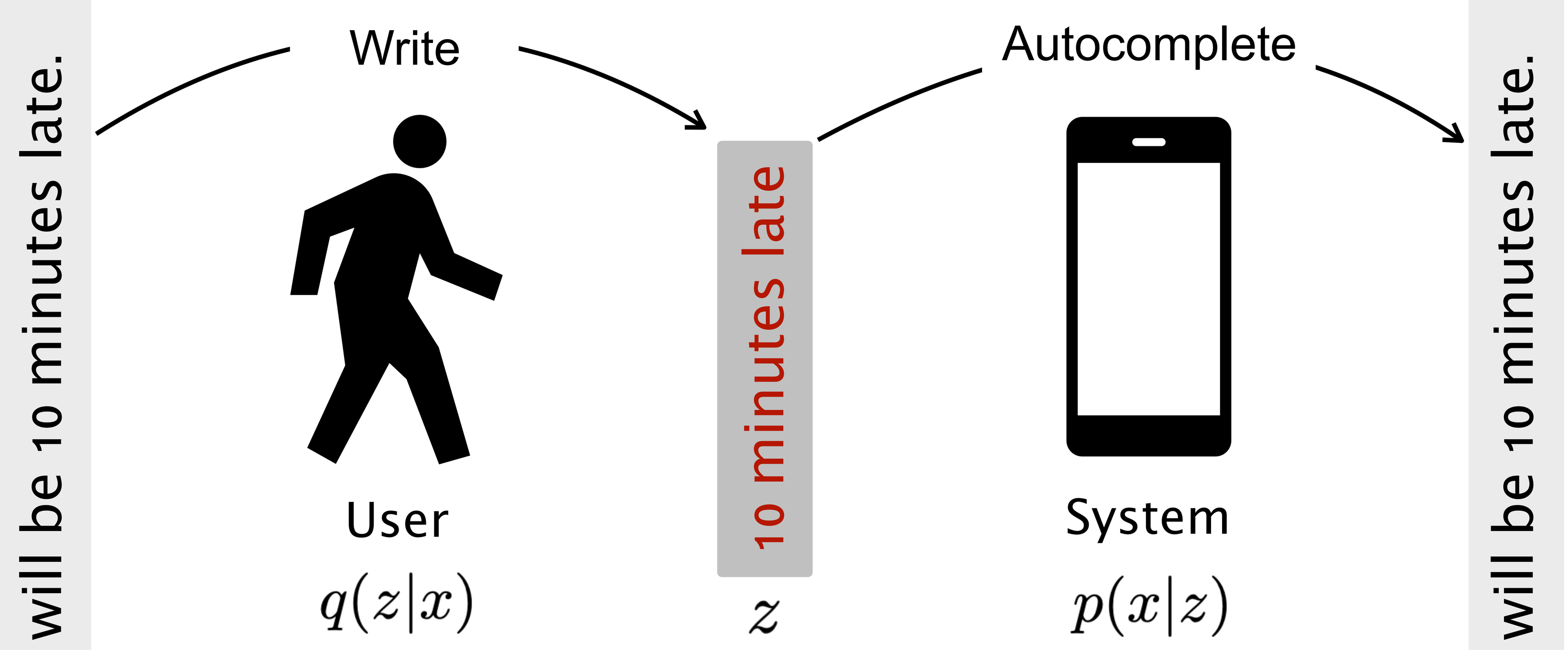
# Learning Autocomplete Systems as a Communication Game

Mina Lee, Tatsunori Hashimoto, Percy Liang
{minalee, thashim, pliang}@stanford.edu

## Task

Given a small number of **keywords**
generate a **sentence**




**Keywords** are a subsequence of the **target sentence**
(potentially noncontiguous)

$x$ $x$

## Approach

Frame it as a **cooperative communication game**
and develop an **unsupervised learning** approach

### Two competing goals

$\mathrm{cost}(x, \alpha) = \mathbb{E}_{q_\alpha(z|x)}[\#\mathrm{tokens}(z)]$ Number of keywords

$\mathrm{loss}(x, \alpha, \beta) = \mathbb{E}_{q_\alpha(z|x)}[-\log p_\beta(x|z)]$ Reconstruction error

### Modeling with autoencoders

**Encoder (User)** Encode x into z by keeping a subset of tokens

**Decoder (System)** Predict distribution over x conditioning on z

### Multi-objective optimization

**Linear** $\min_{\alpha,\beta} \mathbb{E}[\mathrm{cost}(x,\alpha)] + \lambda \mathbb{E}[\mathrm{loss}(x,\alpha,\beta)]$ ✗

**Constrained** $\min_{\alpha,\beta} \mathbb{E}[\mathrm{cost}(x,\alpha)]$
subject to $\mathbb{E}[\mathrm{loss}(x,\alpha,\beta)] \leq \epsilon$ ✓

### Optimization using policy gradient

**Loss function** Lagrangian of constrained objective

$$\min_{\alpha,\beta} \max_{\lambda \geq 0} J(\alpha, \beta, \lambda)$$

$$J(\alpha, \beta, \lambda) = \mathbb{E}[\mathrm{cost}(x, \alpha)] + \lambda(\mathbb{E}[\mathrm{loss}(x, \alpha, \beta)] - \epsilon)$$

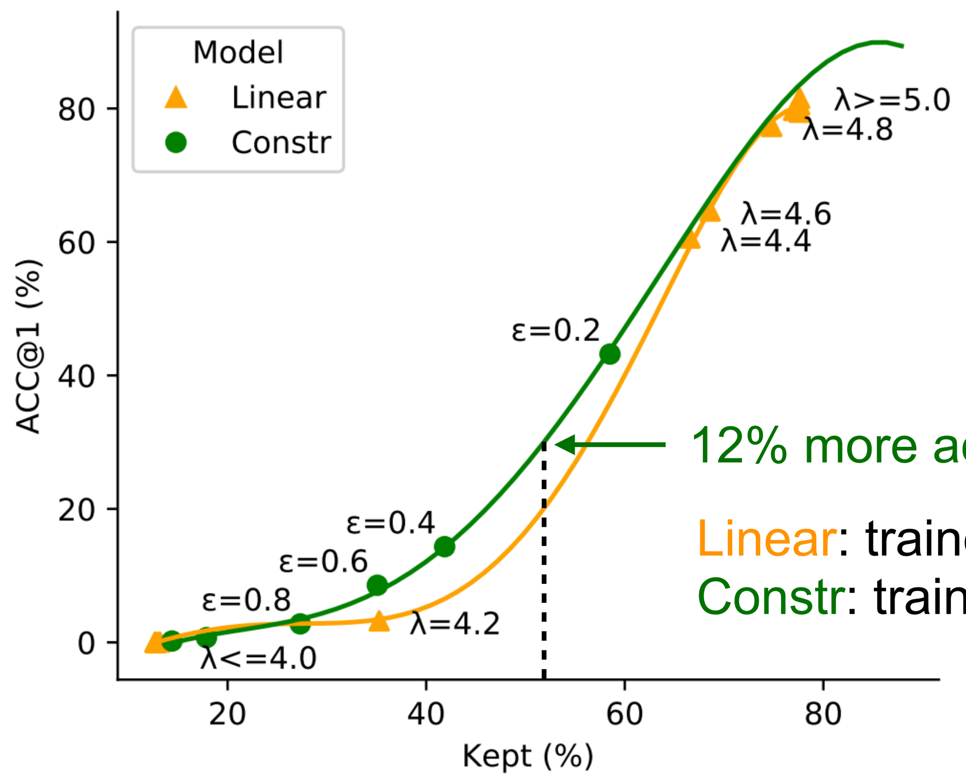
**Policy gradient** REINFORCE with baseline using single sample

$$\nabla_\alpha J(\alpha, \beta, \lambda) = \mathbb{E}[\mathbb{E}_{q_\alpha(z|x)}[\nabla_\alpha \log q_\alpha(z \mid x) \cdot G(x, z)]]$$

$$G(x, z) = \underbrace{\#\mathrm{tokens}(z)}_{\text{per-example cost}} + \lambda(\underbrace{-\log p_\beta(x|z)}_{\text{per-example loss}} - \epsilon)$$

## Experiments

### Why constrained objective?



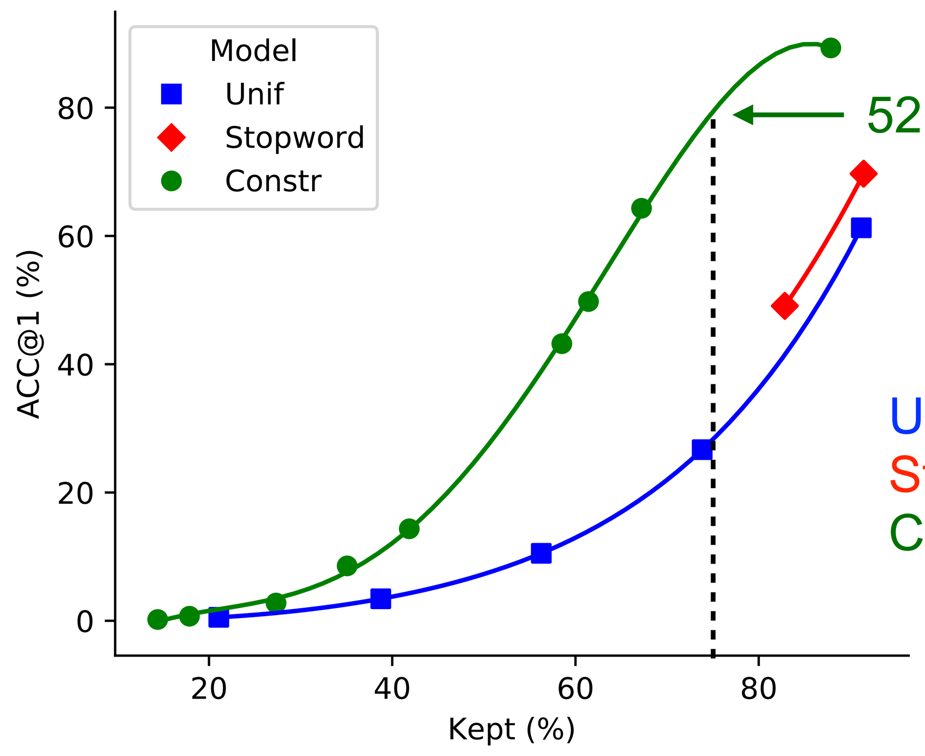

12% more accurate

Linear: trained w/ linear objective
Constr: trained w/ constrained objective

| Linear | Constr |
|---|---|
| Unstable training | Stable training |
| Suboptimal tradeoff | Better tradeoff |
| Arbitrary value for lambda | Interpretable value for epsilon |

### Why joint training of encoder and decoder?

**Better tradeoff** between **cost** and **loss**




52% more accurate

Unif: randomly select keywords
Stopword: drop only stopwords
Constr: learn to select keywords

Selecting random keywords results in bad decoder
Jointly trained encoder drops more than just stopwords

### Correlation with part-of-speech

**Higher kept rates** for **content words**

| Part-of-Speech | Examples | Kept (%) |
|---|---|---|
| **Determiner** | the, a, this | 4 |
| **Conjunction** | and, but | 5 |
| **Pronoun** | it, you, we | 10 |
| | ... | |
| **Verb** | love, recommend | 28 |
| **Adverb** | very, pretty | 35 |
| **Adjective** | delicious | 36 |
| **Noun** | service, food | 40 |

### User study

This keyword-based autocomplete can **save time by nearly 50%** compared to **fully typing sentences**

**Time for typing**
Keywords: **3.85 seconds**
Sentences: 5.76 seconds

**Accuracy**
Paraphrased: **81%**
Exact match: 18%

MaskGAN: Better Text Generation via Filling in the ______ (Fedus et al., 2018)
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al., 2019)
BottleSum: Unsupervised and Self-supervised Sentence Summarization using the Information Bottleneck Principle (West et al., 2019)