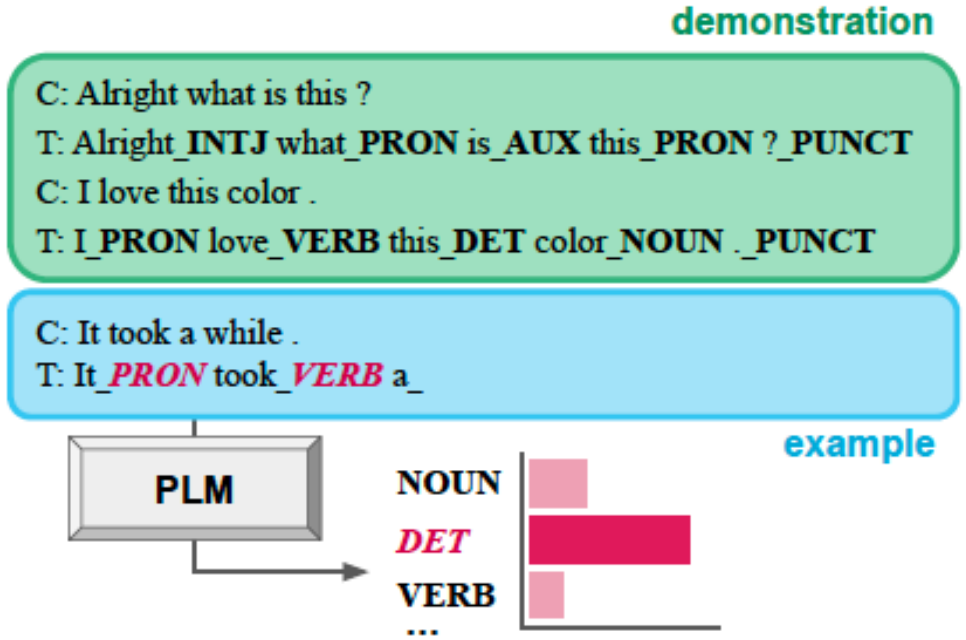
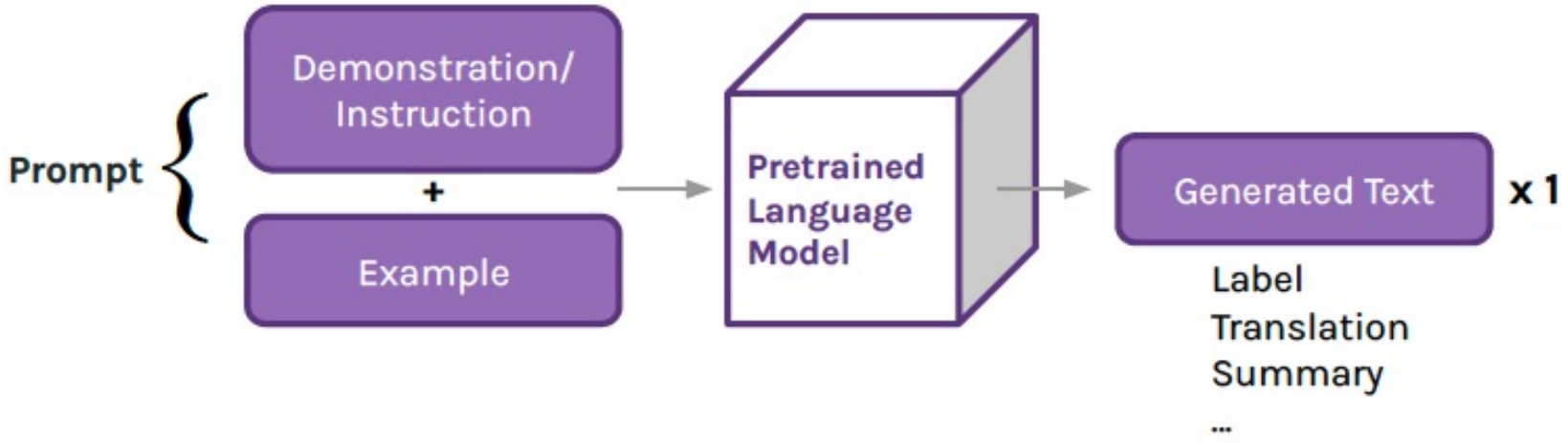
Prompting Language Models for Linguistic Structure

Terra Blevins, Hila Gonen, Luke Zettlemoyer
University of Washington
読み手: 三輪誠 (豊田工業大学)
※図表は論文，著者のスライドより引用

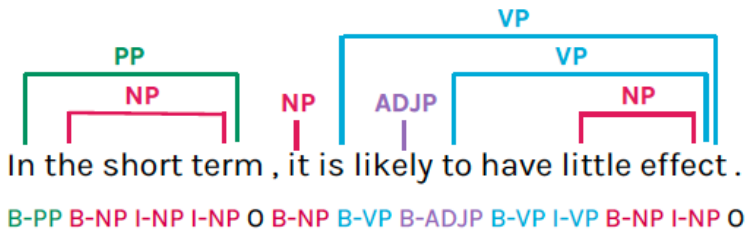
まとめ

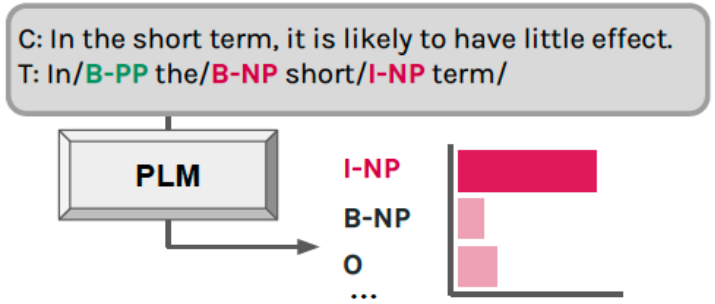
- 系列タグづけのための**構造化プロンプト**の提案
- ラベルの違いや事前学習データによる影響を調査





目的





- 分類・要約・生成などの様々な言語的な知識が必要なタスクで コンテキスト内学習（プロンプト）が成功

➔大規模言語モデルが十分な言語的知識をモデルが持っている, という仮説が立つが，評価はできていない

- 現在のプロンプトは複雑な言語的な構造を扱えない

- 複雑な言語的な構造（系列）を扱える手法の提案
- 手法を用いた大規模言語モデルの振る舞いの評価・調査

構造化プロンプト

- 従来のプロンプト
  - 入力に対して，出力を一度に生成
- 構造化プロンプト
  (structured prompting)
  - 単語ごとに入力・予測を繰り返す
  1. モデルに単語を入力し，出力ラベルを予測
  2. モデルに予測したラベルを入力し，1に戻る

  - 単語は生成しない
  - 単語が複数サブワードに分かれる場合は対数尤度の平均を利用
  - 出力はラベルに限定

2 Structured Prompting of Pretrained Language Models

We propose a sequential method for performing sequence tagging with PLMs via in-context learning, which we refer to as *structured prompting* (Figure 1). The model is given k (context, tagged sequence) pairs as the task demonstration and the example sentence to be labeled. The model then iteratively tags the words in the example with constrained decoding over a fixed set of labels.

More specifically, given a set of labels L and an input sequence c containing k demonstration pairs as well as the full text of the example sentence $S = s_0, ..., s_n$, at each time step t the language model M encodes $[c; s_t]$ and labels s_t with $\hat{\ell}_t = \underset{\ell \in L}{\text{argmax}}\, P_M(\ell|c, s_t)$. We then update the input sequence by appending the current word s_t and the predicted label $\hat{\ell}_t$ to the end of c. Multi-token labels are scored with the average log-likelihood over all tokens $P_M(\ell|c) = \frac{1}{|\ell|}\sum_{i=0}^{|\ell|} P_M(y_i|c, y_0, ..., y_{i-1})$, where y_j is the jth subword token in ℓ.

This approach to in-context learning tags an entire sequence with a single pass over the context. It also allows the model to condition on past predictions while labeling the current word. As we demonstrate in Section 4, these features allow us to apply large autoregressive language models to a broad class of core NLP tasks in a few-shot manner.

問題設定

C: Alright what is this ?
T: Alright_INTJ what_PRON is_AUX this_PRON ?_PUNCT
C: I love this color .
T: I_PRON love_VERB this_DET color_NOUN ._PUNCT

- Few-shot学習
  - プロンプトのフォーマット
    - C: Context (元の文), T: Tagged (タグづけ結果)
    - 事前実験において，識別子 (_) などの影響は小さいが,
      Tから単語を除くと，POSで84%, NERで79%性能が低下 (GPT-J-6B)
- タスク
  - 品詞タグづけ: GUMコーパス (English Universal Dependency)
    - 17ラベル (ADJ, ADP, …)
  - Chunking: CoNLL 2000
    - 23ラベル (B-NP, I-NP, B-VP,…, O)
  - NER: CoNLL 2003
    - 9ラベル (B-PER, I-PER, …, O)

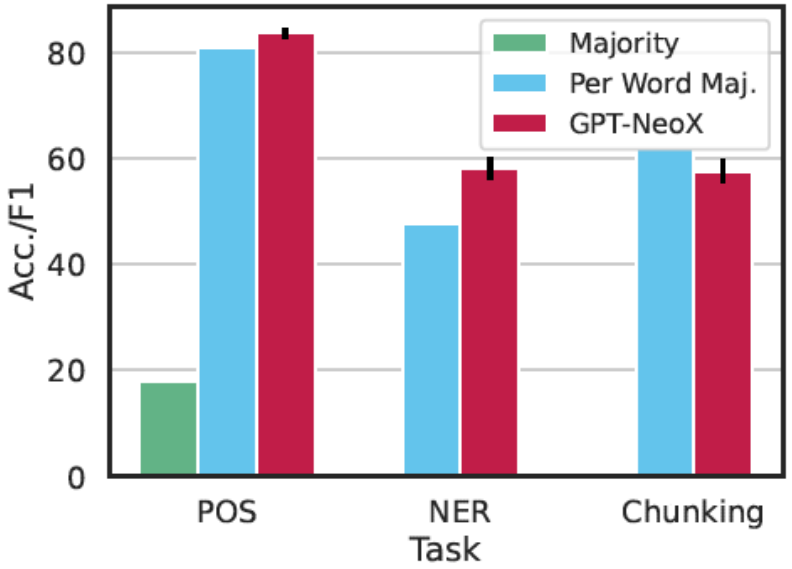
実験設定

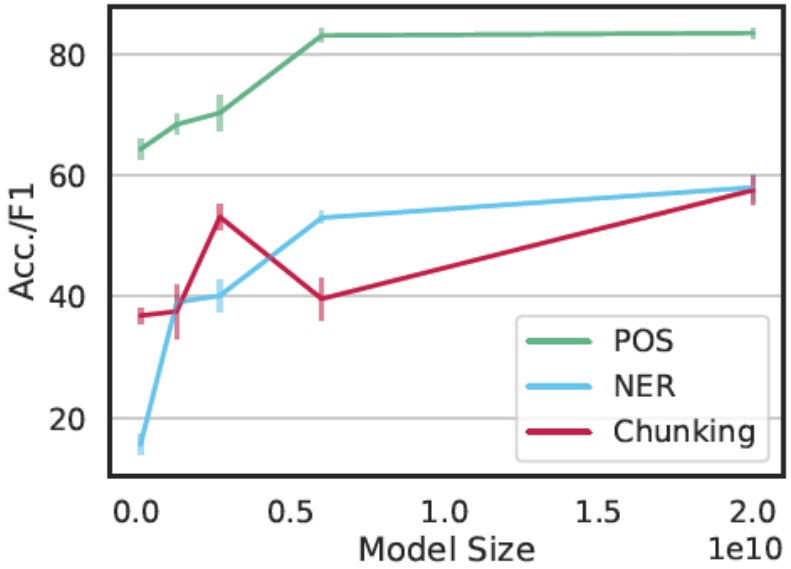
- モデル
  - GPT-NeoX
    - 125M, 2.7B, 6.7B, **20B**
  - GPT-J-6B
  - GPT-3 (APIを利用)
    - GPT-Curie (6B), GPT-Davinci (175B; 従来のプロンプトのみ)

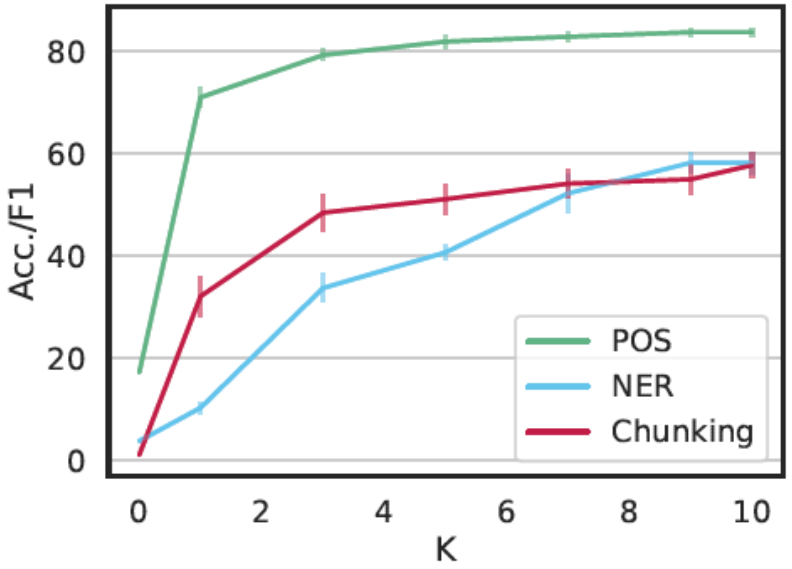
  ※言及がなければ，GPT-NeoX-20Bで評価
- 評価
  - GreedyにBIOの制約を考慮しながらデコード
  - （できるだけ未知ラベルがないように）10例までデモを見せて，生成.
  - 1000例で評価．5回の評価を平均.

Few-shot学習での評価

- ベースラインであるmajority vote（単語レベル, 全体）よりも高性能
- モデルサイズ，Few-shotの事例数 (K) に合わせて性能向上
  - NERでは特に事例数に敏感







構造化プロンプトの効果

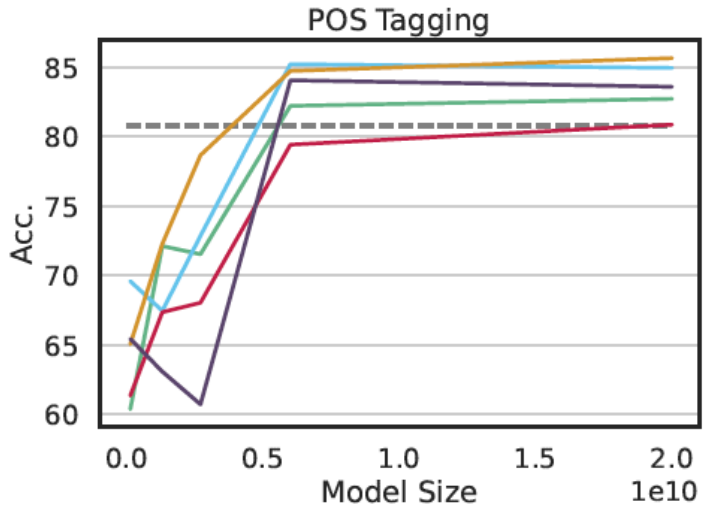
- 構造化プロンプトを利用した小さいGPT-J, GPT-Curieの方が, 通常プロンプトのGPT-Davinciよりも高性能
  - k: few-shotの事例数, Acc: 正解率, SE: standard error
- GPT-Davinciはフォーマットエラーも多いが, フォーマットエラーを除いても72.85% (k=5), 78.04% (k=10)

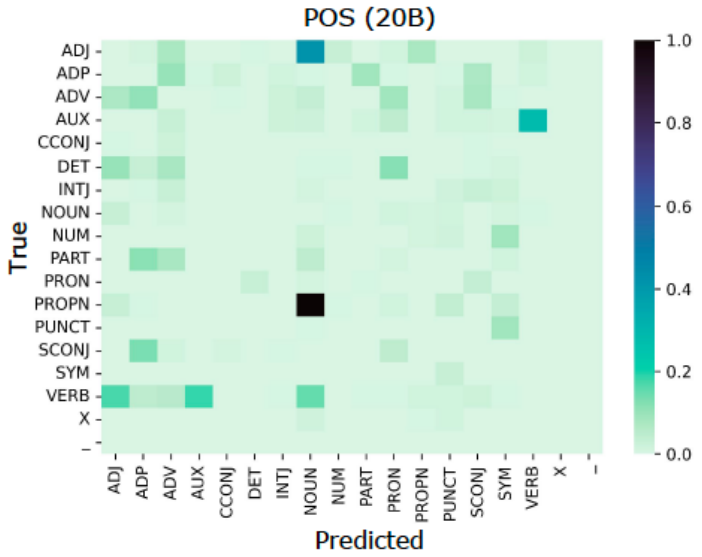
Size	Model	k	Acc.	SE
~6B	GPT-J*	5	79.01	2.95
	GPT-Curie	5	66.27	0.46
~175B	GPT-Davinci†	5	59.65	2.84
	GPT-Davinci†	10	65.90	1.34

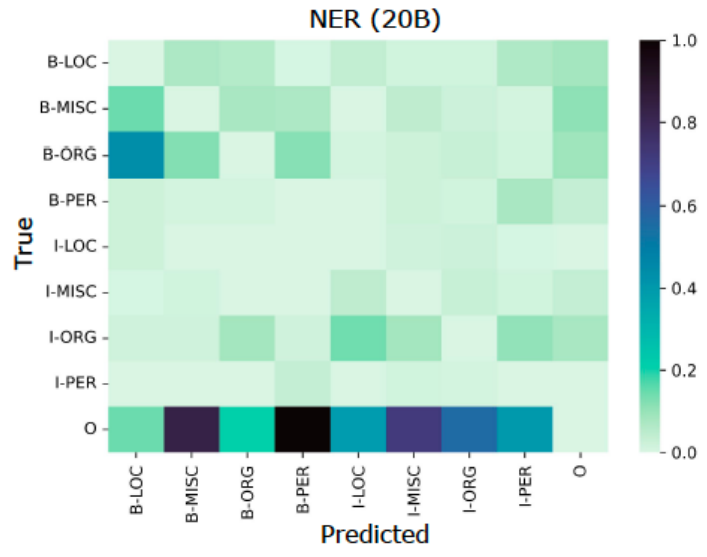
POSタグづけにおける構造化プロンプト (6B) と通常のプロンプト (175B) の比較

エラー解析

- 事例（デモ）で性能が異なる（Chunkingでは差が大きい）
- POSでは近いラベルで間違っているが，NERはよくわからない
  - 間違いの傾向はデモに大きくは依らない（スピアマン相関係数大）
    - POS, NER, Chunkingのρ: 0.77, 0.84, 0.88 (20B) → 0.71, 0.64, 0.66 (2.7B)
      ➔ モデルが大きいほうがロバスト

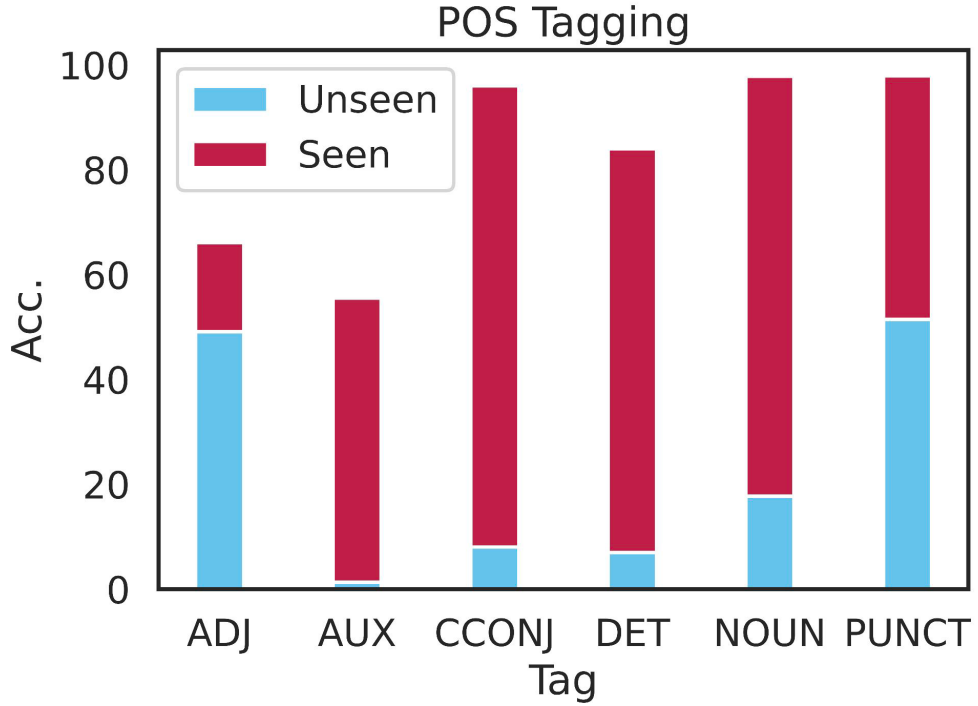


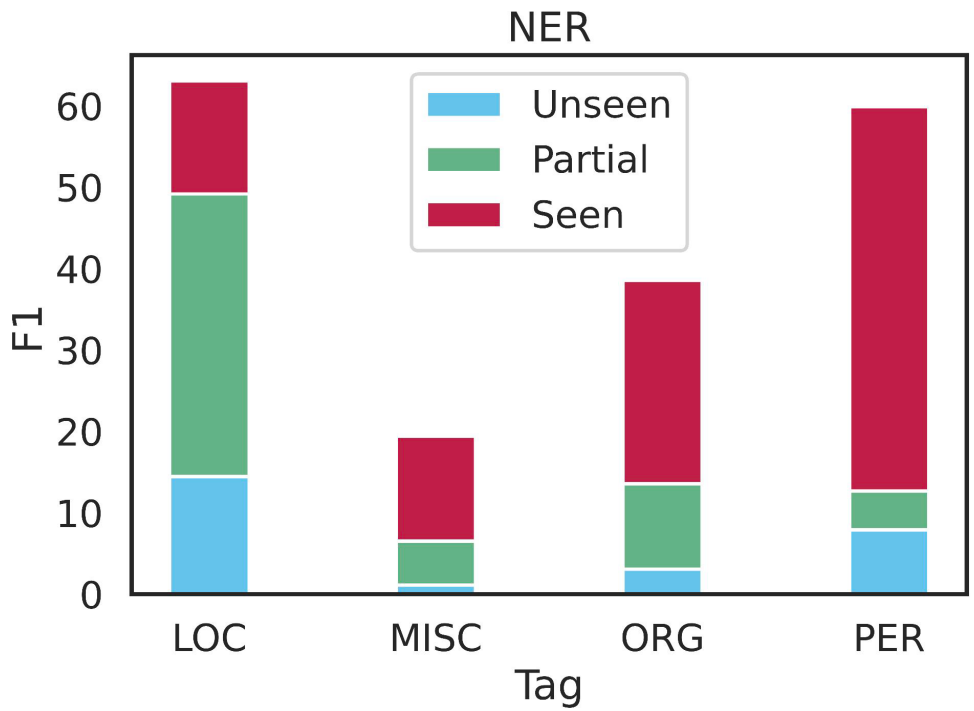




Few-shotとZero-shotの違い

- Zero-shotで50%以上解けるものも解けないものもある
  ➔ すでにラベルを知っている
- 性能向上の幅もタイプによる





※PartialはBIOのBIのどちらかのみ

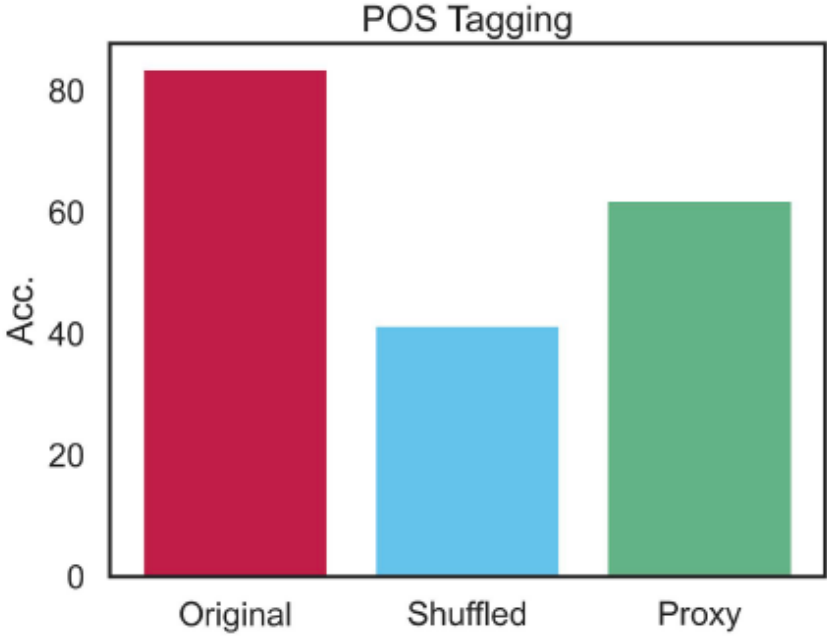
ラベルの表層の影響の調査

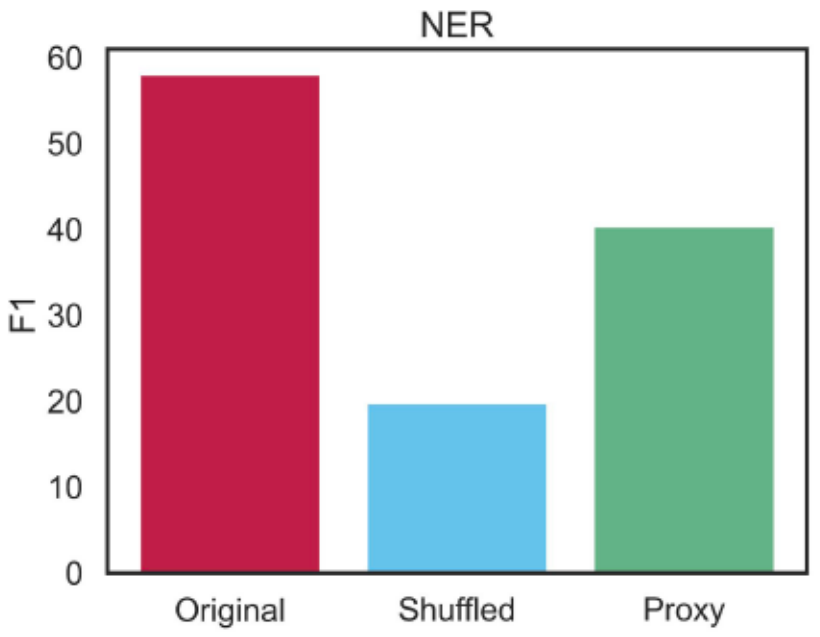
- Shuffled: ラベル名をシャッフル
- Proxy: 連続した数字 (11~) でタイプ名を置き換え
  - NERの場合はB-11など

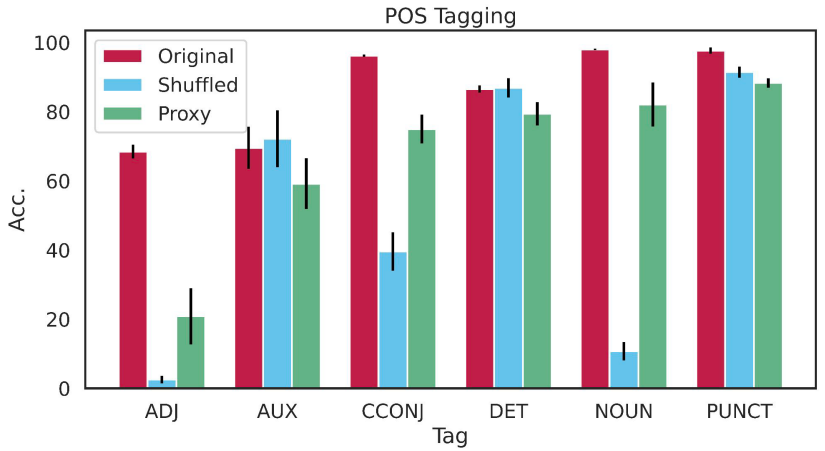
	The	cat	is	a	…
Original Labels	DET	NOUN	AUX	DET	
Shuffled Labels	PUNCT	ADV	PROPN	PUNCT	
Proxy Labels	13	18	26	13	

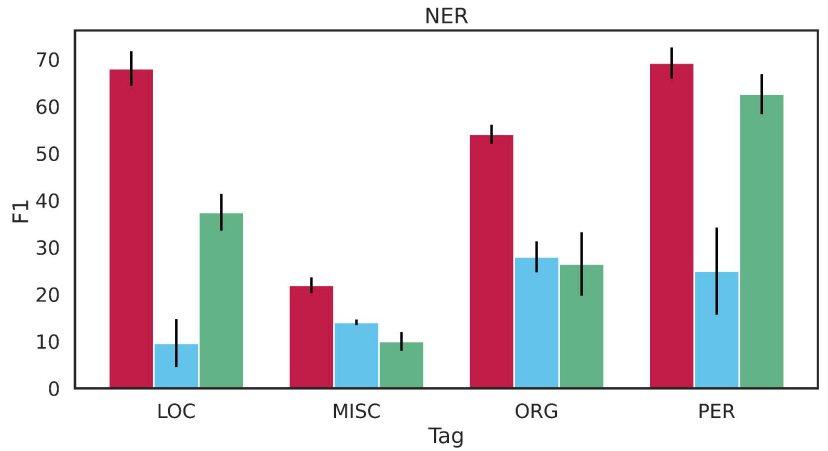
ラベルの表層の影響の調査結果

- Shuffled
  - 61.4%のエラーでシャッフル前のクラスを予測
  - 知っているものはコンテキスト内学習での変更が難しい
- Proxy
  - ProxyはShuffledに比べると予測できる









事前学習データの調査

- 事前学習データ（Pile）中でのタスクラベルの調査
  - GitHubにUDのコーパスはあるが，GUMコーパスと一致する文はなかった．CCONJの77%は英語以外．
  - NERは100例を見たが，多くはCoNLLフォーマット以外，英語以外，間違ったラベル
- タスク説明文やコードも見つかったが，テストデータのリークはなかった

Label	Freq.	Task Stats	Example Contexts
POS Tagging		**UD Format**	
NOUN	360k	9.29%	The 10 most frequent relations where parent and child node agree in ‘Polarity‘: <tt>**NOUN** —-> ADJ</tt> (2; 100%) (GitHub)
CCONJ	22k	23.48%	13 \t und \t und \t **CCONJ** \t KON \t _ \t 14 \t cc (GitHub)
DET	1.53M	0.72%	**DET**: determiner, e.g. a, an, the \n INTJ: interjection, e.g. psst... (StackExchange)
NER		**Relevant?**	
B-PER	5,655	26/100	Bacterial pellets were lysed in 10 ml **B-PER** Bacterial Protein Extraction Reagent... (PubMed)
I-LOC	2,197	43/100	y = np.asarray("B-PER O O B-LOC **I-LOC** O B-ORG".split()) (StackExchange)
B-ORG	2603	80/100	*I-PER* label usually follows *B-PER* and *I-PER*, but it cannot follow ***B-ORG*** or *I-ORG*. (Arxiv)
I-MISC	907	76/100	My(O) favorite(O) book(O) is(O) harry(B-MISC) potter(**I-MISC**)... (StackExchange)

事前学習データの影響の軽減

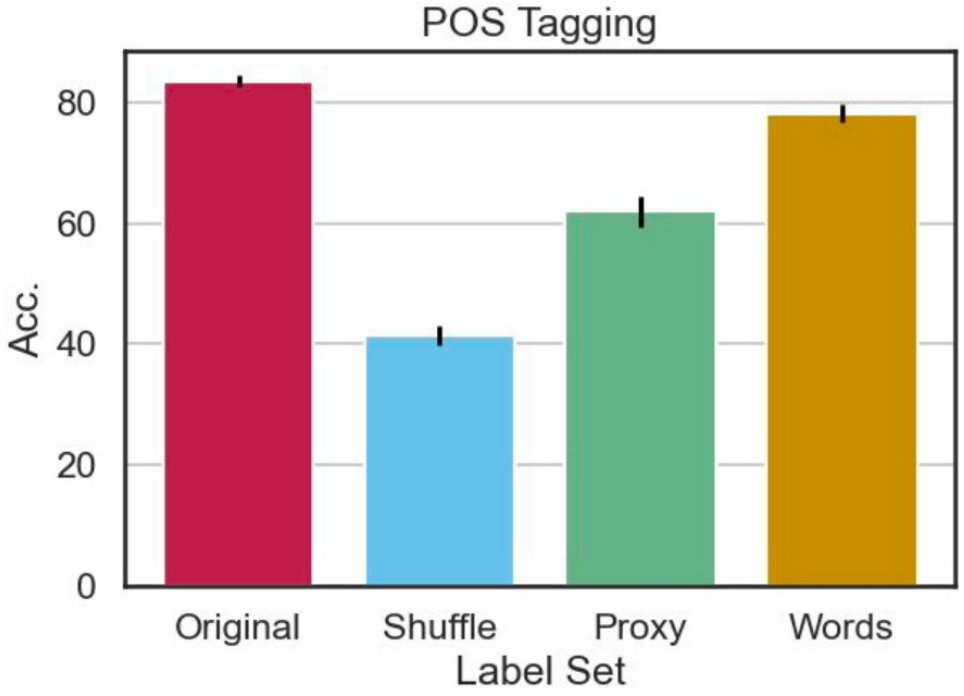
- Words as labels
  - ラベル名を意味のある英語に変換
    - ADJ → adjective, B-LOC → B-location

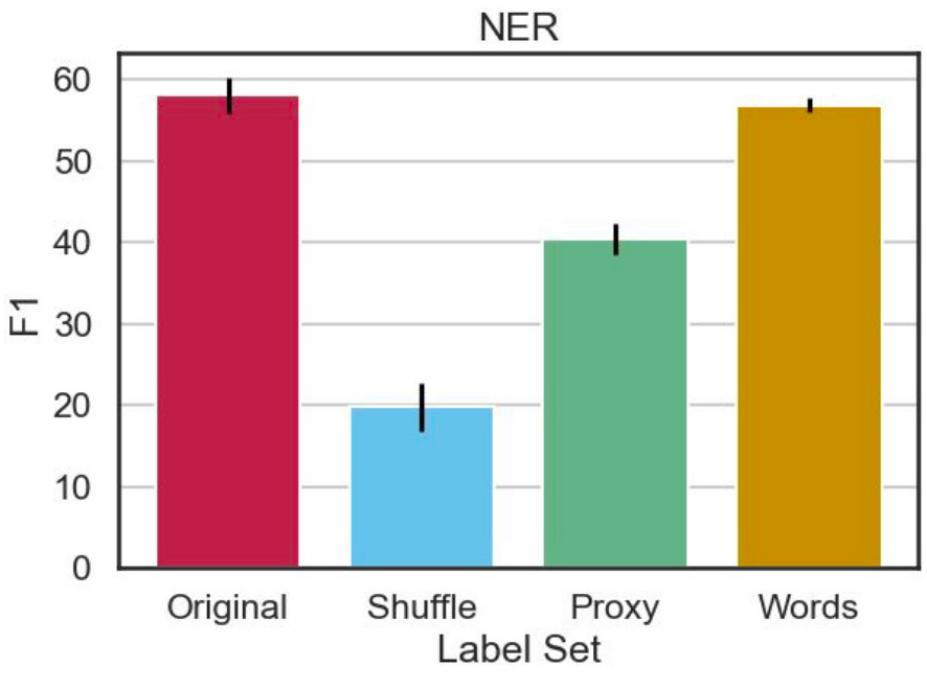
	The	cat	is	a	…
Original Labels	DET	NOUN	AUX	DET	
Shuffled Labels	PUNCT	ADV	PROPN	PUNCT	
Proxy Labels	13	18	26	13	
➡ **Words** as Labels	determiner	noun	auxiliary	determiner	

事前学習データの影響の軽減の結果

- 性能の低下は限定的 (v.s. Original)
  - 特に，NERでは標準偏差内

➔ 単にラベル名を覚える以上の汎化をしている





まとめ

- 大規模言語モデルにおける系列タグづけのための構造化プロンプトを提案
- 事前学習データ内のタグ付きデータによって，タスクに関する事前知識を持っているが，コンテキスト内学習ではそれ以上の汎化をしている (Proxy, Words)
  ➔ 覚える（Memorization）以上の言語構造への知識を持っている