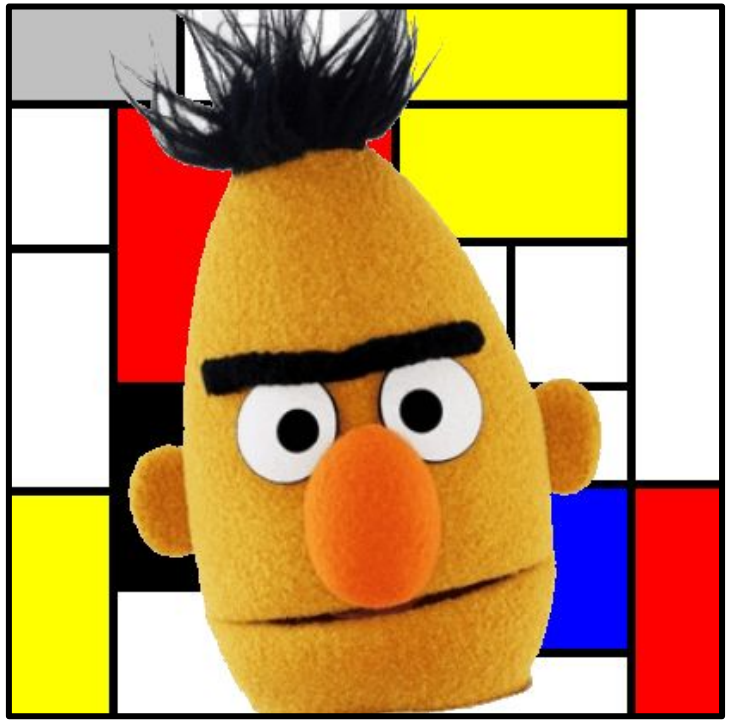
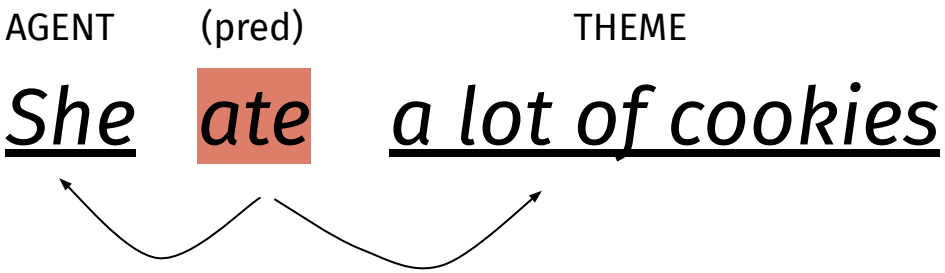
# BERT for end-to-end FrameNet SRL

**Gosse Minnema**
CL Reading Group
April 17, 2020



# Background

**Semantic Role Labeling (SRL):**
capture semantic dependencies between predicates and arguments

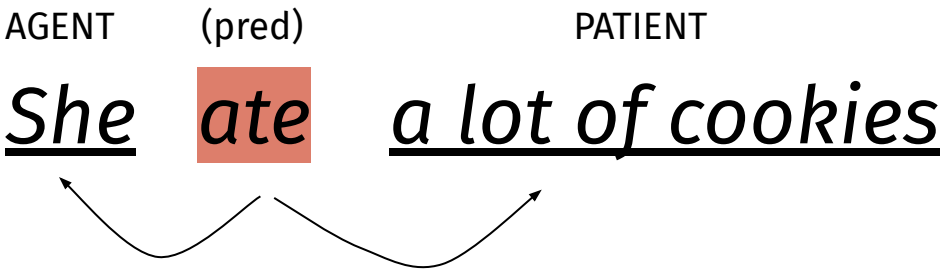


# Background

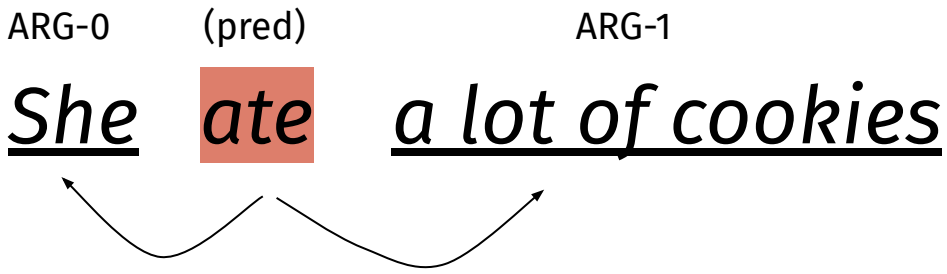
How do you label semantic dependencies?

⇒ **different styles of semantic role labeling**

**VerbNet-style**: general semantic roles, shared between all predicates



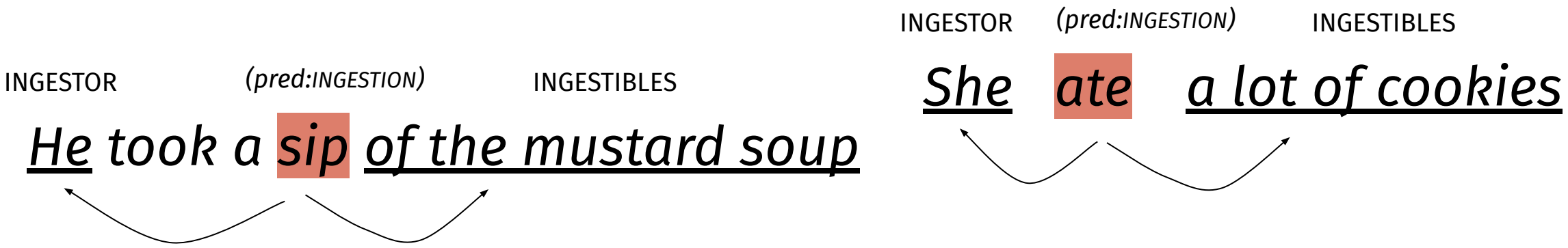

**PropBank-style**: every predicate has its own set of semantic roles



# Background

**FrameNet-style semantic role labeling:**

- sets of semantic roles are shared across “frames”
- frames group together a set of related predicates



# Background

**Why are frames interesting?**

- In general: connect semantic roles to general conceptual knowledge
  - “frames” are also used in AI outside of NLP
- Lexicography: define word meanings in a specific conceptual context
  - e.g. a “sip” can only be defined in context of eating/drinking
- NLP: can you learn to make the right level of generalization?

# The problem

**FrameNet-style SRL pipeline:**

1) `targetID` - identify predicates

   *he saw the cookies and ate them ⇒ "saw", "ate"*

2) `frameID` - for each target, find the correct frame

   *"saw" ⇒ PERCEPTION_EXPERIENCE; "ate" ⇒ INGESTION*

3) `argID` - for each target+frame, assign semantic roles to its dependents

   *PERCEPTION_EXPERIENCE ⇒ PERCEIVER ("he"), PHENOMENON ("the cookies")*

   *INGESTION ⇒ INGESTOR ("he"), INGESTIBLES ("them")*

# The problem

Two observations:

- **The pipeline is not really a pipeline**
  - `frameID` ⇒ `argID`: what is the set of possible roles for a predicate?
  - `argID` ⇒ `frameID`: which frame fits best with the arguments of a predicate?

- **`argID` is an underestimated problem**
  - Most literature focuses on `argID`, but `frameID` seems to be main bottleneck for cross-domain generalization *(Hartmann et al., EACL 2017)*

# The problem

**The idea:** can we use BERT as a basis for a new FrameNet-SRL model that…

1) Does `frameID` + `argID` at the same time?
   ⇒ SRL as a sequence labeling problem

2) Improves performance on `frameID`?

# The problem

Why BERT?

- **Transfer learning:** FrameNet training corpus is small (<3000 sentences), so any pre-training is likely to be helpful
- **Context:** `frameID` is (similar to WSD problems) very context-dependent, so contextualized representations are useful
- **Syntax:** most/all previous FrameNet SRL models rely on dependency parses in some way; BERT can capture dependency structure implicitly

# Designing a sequence labeler

**What do we want as output?**

| | |
|---|---|
| Adherence | T:Compliance |
| to | B:Compliance:Norm |
| eating | I:Compliance:Norm\|T:Ingestion |
| Islamic | I:Compliance:Norm\|B:Ingestion:Ingestibles\|B:Food:Type |
| Halal | I:Compliance:Norm\|I:Ingestion:Ingestibles\|I:Food:Type |
| food | I:Compliance:Norm\|I:Ingestion:Ingestibles\|T:Food\|B:Food:Food |

**⩽ 1 frameID label per token**

# Designing a sequence labeler

**What do we want as output?**

| | |
|---|---|
| Adherence | T:Compliance |
| to | B:Compliance:Norm |
| eating | I:Compliance:Norm\|T:Ingestion |
| Islamic | I:Compliance:Norm\|B:Ingestion:Ingestibles\|B:Food:Type |
| Halal | I:Compliance:Norm\|I:Ingestion:Ingestibles\|I:Food:Type |
| food | I:Compliance:Norm\|I:Ingestion:Ingestibles\|T:Food\|B:Food:Food |

**⩽ 1 `frameID` label per token** **any number of `argdID` labels (IOB:frame:role) per token**

# Designing a sequence labeler

**What do we want as output?**

| | |
|---|---|
| Adherence | T:Compliance |
| to | Norm |
| eating | Norm\|T:Ingestion |
| Islamic | Norm\|Ingestibles\|Type |
| Halal | Norm\|Ingestibles\|Type |
| food | Norm\|Ingestibles\|T:Food\|Food |

***Problem: huge label space!***

Simplification: assume we can reconstruct IOB and frame tags in `argID`

# Designing a sequence labeler

**How to represent the labels?**

`argID+frameID` together: sparse, binary vectors

$$\begin{pmatrix} \texttt{frameID}_1, \texttt{frameID}_2, \ldots \texttt{frameID}_n, \texttt{argID}_1, \texttt{argdID}_2, \ldots \\ \texttt{argID}_n \end{pmatrix}$$

`argID` alone: sparse, binary vectors

$$\begin{pmatrix} \texttt{argID}_1, \texttt{argdID}_2, \ldots \texttt{argID}_n \end{pmatrix}$$

***Q: is there a smarter way to do this?***

# Designing a sequence labeler

**How to represent the labels?**

`frameID` alone: sparse, binary vectors

$$\begin{pmatrix} \texttt{frameID}_1, & \texttt{frameID}_2, & \ldots & \texttt{frameID}_n \end{pmatrix}$$

OR: use frame embeddings

- best approach so far: average GloVe embeddings of all possible predicates of each frame
  e.g.: `INGESTION = mean(eat, sip, drink, ingest, …)`

# Preliminary results

| system | frameID | | | argID | | |
|---|---|---|---|---|---|---|
| | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ |
| Open-SESAME *(Swayamdipta et al. 2017)* | **0.70** | 0.68 | 0.69 | 0.54 | 0.41 | 0.47 |
| BERT ⇒ `frameID+argID` | 0.57 | 0.71 | 0.63 | 0.44 | **0.61** | **0.51** |
| BERT ⇒ `argID` | - | - | - | 0.43 | 0.52 | 0.47 |
| BERT ⇒ `frameID` (sparse) | 0.67 | **0.75** | **0.71** | - | - | - |
| BERT ⇒ `frameID` (embedding) | 0.69 | 0.71 | 0.70 | - | - | - |

**Some observations:**

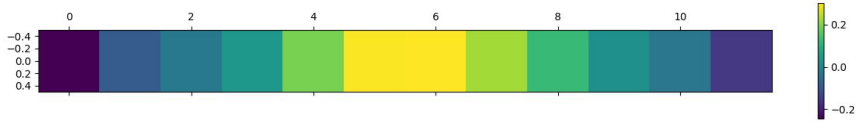
- New SOTA for `frameID` and `argID`, but not at the same time
- `argID` benefits from `frameID` but not vice versa
- `argID`-only system performs at SOTA level without frame as input

# Preliminary results

## "BERT re-discovers the FrameNet pipeline" (???)

| BERT layers | frameID | | | argId | | |
|---|---|---|---|---|---|---|
| | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ |
| Learned layer mix | **0.65** | **0.71** | **0.68** | 0.39 | **0.60** | 0.48 |
| BERT ⇒ `frameID`, layer 6 | 0.51 | **0.71** | 0.60 | 0.38 | 0.56 | 0.45 |
| BERT ⇒ `frameID`, layer 8 | 0.54 | 0.70 | 0.61 | 0.40 | 0.55 | 0.46 |
| BERT ⇒ `frameID`, layer 10 | 0.57 | **0.71** | 0.63 | 0.44 | 0.61 | **0.51** |
| BERT ⇒ `frameID`, layer 12 | 0.46 | **0.71** | 0.56 | **0.45** | 0.59 | **0.51** |

**Learned layer mix**
with learned parameters `{gamma, w}`
`mix = gamma * sum(s_k * layer_k)`
where `s = softmax(w)`



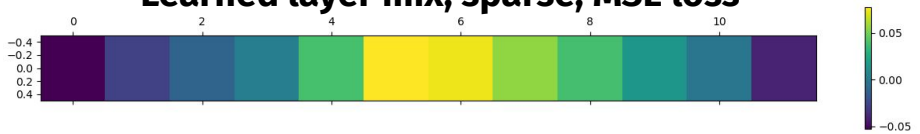

- Last layer gives good `argID`, bad `frameID` results
- Learned layer mix gives better `frameID` results than any individual layer

# Preliminary results

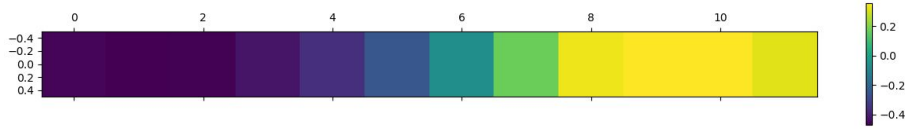
## "BERT re-discovers the FrameNet pipeline" (???)

| BERT layers | `frameID` (sparse) | | | `frameID` (emb.) | | |
|---|---|---|---|---|---|---|
| | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ | $R_{dev}$ | $P_{dev}$ | $F1_{dev}$ |
| BERT ⇒ `frameID`, learned layer mix | 0.65 | **0.76** | 0.70 | 0.69 | **0.70** | 0.69 |
| BERT ⇒ `frameID`, layer 6 | **0.68** | 0.73 | 0.70 | 0.69 | 0.65 | 0.67 |
| BERT ⇒ `frameID`, layer 8 | 0.67 | 0.75 | **0.71** | 0.70 | **0.70** | **0.70** |
| BERT ⇒ `frameID`, layer 10 | 0.65 | 0.75 | 0.70 | **0.71** | 0.69 | **0.70** |
| BERT ⇒ `frameID`, layer 12 | 0.64 | 0.73 | 0.68 | 0.70 | 0.69 | 0.69 |

**Learned layer mix, sparse, MSE loss**




**Learned layer mix, embedding, COS loss**




- Layer mix parameters: sparse uses middle layers, embedding uses late layers
- Individual layer results are not so clear
- Embedding predictions have more balance between P and R

# Next steps

- **Multi-task learning:** predict separate labels for `frameID`, `argID`
- **Embedding role embeddings** - but how to deal with multiple labels?
- **Study the "pipeline" BERT learns**
  - Is this feasible?
  - How to do it?