

# Bringing Scientific Papers to Life

## ICDAR 2023 Keynote

Marti Hearst  
UC Berkeley

In collaboration with the Allen Institute of AI

Berkeley SCHOOL OF INFORMATION



# Scholarly Papers Contain The World's Scientific Knowledge

Publications are growing exponentially  
And papers can be challenging to understand

# Scholarly Papers Contain The World's Scientific Knowledge

How can we make that knowledge more  
widely and broadly accessible?

# Scholarly Papers Contain The World's Scientific Knowledge

How can we make that knowledge more  
widely and broadly accessible?

Our approach: Use NLP, HCI, and  
document analysis to augment papers.



# Talk Outline

**1. Three Motivating Demos**

**2. AI2's Semantic Reader Project**

**3. NLP and Document Analysis Techniques  
Including GPT-4 Experiments**

# Talk Outline

## 1. Three Motivating Demos

## 2. AI2's Semantic Reader Project

## 3. NLP and Document Analysis Techniques Including GPT-4 Experiments

# 1. Three Motivating Demos

# ScholarPhi

Scim

# PaperPlair



Figure 2: Syntactically-informed self-attention for the query word *sloth*. Attention weights heavily weight the token’s syntactic *isw*, in a weighted average over the tokens using  $V_{\text{parse}}$ . The other attention heads are usual, and the attended representations from heads are concatenated and projected to the feed-forward layer to produce the syntactically-informed representation for *sloth*.

trained word embeddings, but we also experiment with supplying pre-trained ELMo representations combined with task-specific learned parameters, which have been shown to substantially improve performance of other SRL models (Peters et al., 2018). For experiments with gold predicates, we concatenate a predicate indicator embedding  $p_i$  following previous work (He et al., 2017).

We project<sup>2</sup> these input embeddings to a representation that is the same size as the output of the self-attention layers. We then add a positional encoding vector computed as a deterministic sinusoidal function of  $f_i$ , since the self-attention has no innate notion of token position.

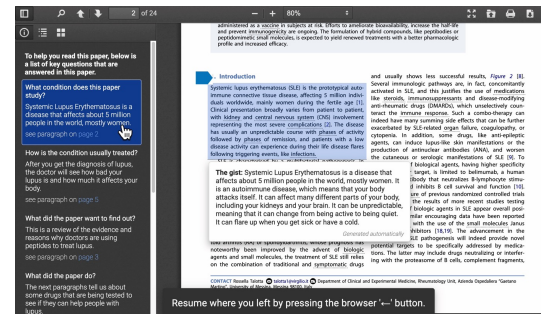
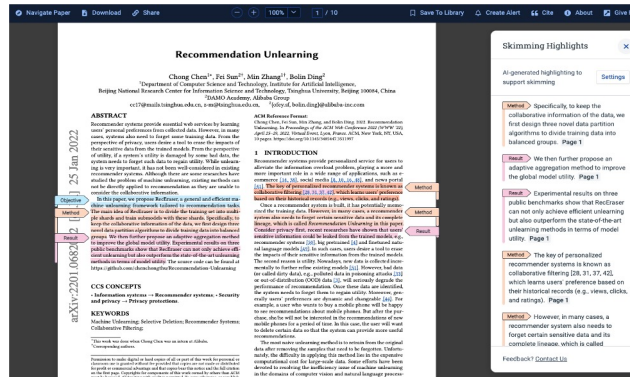
layer normalization

layer

tion function to attend to all of the tokens in the sequence. This self-attention is performed for each token for each head, and the results of the  $H$  self-attentions are concatenated to form the final self-attended representation for each token.

Specifically, consider the matrix  $S^{(j-1)}$  of  $T_{\text{token}}$  token representations at layer  $j - 1$ . For each attention head  $h$ , we project this matrix into distinct

row  
of  
 $M_h^{(j)}$

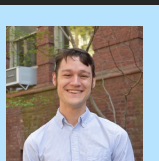


To make scientific papers more understandable

# SCHOLARPHI

Augmenting Scientific Papers with Just-in-Time, Position-Sensitive Definitions of Terms and Symbols

@ACM CHI '21.



Andrew Head



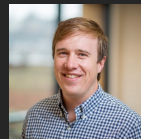
Kyle  
Lo



Dongyeop Kang



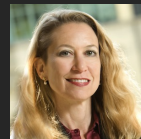
Raymond Fok



Sam Skjonsberg



Daniel S. Weld



Marti A. Hearst

# What makes scientific papers challenging to read?

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>2</sup>, David Weiss<sup>2</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences

University of Massachusetts Amherst

{strubell, pat, mccallum}@cs.umass.edu

<sup>2</sup>Google AI Language

New York, NY

{andor, djweiss}@google.com

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA): a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and SRL. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA can incorporate syntax using merely raw tokens as input, encoding the sequence only once to simultaneously perform parsing, predicate detection and role labeling for all predicates. Syntax is incorporated by training one attention head to attend to syntactic parents for each token. Moreover, if a high-quality syntactic parse is already available, it can be beneficially injected at test time without re-training our SRL model. In experiments on CoNLL-2005 SRL, LISA

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazrafshan and Gildea, 2013).

Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Punyakanok et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models.

Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017)

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>2</sup>, David Weiss<sup>2</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences  
University of Massachusetts Amherst  
{strubell, pat, mcallum}@cs.umass.edu  
<sup>2</sup>Google AI Language  
New York, NY  
{andor, djweiss}@google.com

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA): a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and SRL. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA can incorporate syntax using merely raw tokens as input, encoding the sequence only once to simultaneously perform parsing, predicate detection and role labeling for all predicates. Syntax is incorporated by training one attention head to attend to syntactic parses for each token. Moreover, if a high-quality syntactic parse is already available, it can be beneficially injected at test time without re-training our SRL model. In experiments on CoNLL-2005 SRL, LISA achieves new state-of-the-art performance for a model using predicted predicates and standard word embeddings, attaining 2.5 F1 absolute higher than the previous state-of-the-art on newswire and more than 3.5 F1 on out-of-domain data, nearly 10% reduction in error. On CoNLL-2012 English SRL we also show an improvement of more than 2.5 F1. LISA also out-performs the state-of-the-art with contextually-encoded (ELMo) word representations, by nearly 1.0 F1 on news and more than 2.0 F1 on out-of-domain test.

### 1 Introduction

Semantic role labeling (SRL) extracts a high-level representation of meaning from a sentence, labeling e.g. who did what to whom. Explicit representations of such semantic information have been

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazzafshan and Gildea, 2013). Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Panyakank et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models.

Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017) indicate that many of the errors made by a syntax-free neural network on SRL are tied to certain syntactic confusions such as prepositional phrase attachment, and show that while constrained inference using a relatively low-accuracy predicted parse can provide small improvements in SRL accuracy, providing a gold-quality parse leads to substantial gains. Marcheggiani and Titov (2017) incorporate syntax from a high-quality parser (Kiparwasser and Goldberg, 2016) using graph convolutional neural networks (Kipf and Welling, 2017), but like He et al. (2017) they attain only small increases over a model with no syntactic parse, and even perform worse than a syntax-free model on out-of-domain data. These works suggest that though syntax has the potential to improve neural network SRL models, we have not

and normalize with the softmax function to produce a distinct distribution for each token over all the tokens in the sentence:

$$A_t^{(l)} = \text{softmax}(d_t^{(l)} \odot q_t^{(l)} K_t^{(l)T}) \quad (2)$$

These attention weights are then multiplied by  $V_t^{(l)}$  for each token to obtain the self-attended token representations  $M_t^{(l)}$ :

$$M_t^{(l)} = A_t^{(l)} V_t^{(l)} \quad (3)$$

Row  $t$  of  $M_t^{(l)}$ , the self-attended representation for token  $t$  at layer  $l$ , is thus the weighted sum with respect to  $t$  (with weights given by  $A_t^{(l)}$ ) over the token representations in  $V_t^{(l)}$ .

The outputs of all attention heads for each token are concatenated, and this representation is passed to the feed-forward layer, which consists of two linear projections each followed by leaky ReLU activations (Maas et al., 2013). We add the output of the feed-forward to the initial representation and apply layer normalization to give the final output of self-attention layer  $l$ , as in Eqn. 1.

### 2.2 Syntactically-informed self-attention

Typically, neural attention mechanisms are left on their own to learn to attend to relevant inputs. Instead, we propose training the self-attention to attend to specific tokens corresponding to the syntactic structure of the sentence as a mechanism for passing linguistic knowledge to later layers.

Specifically, we replace one attention head with the deep bi-affine model of Drazot and Manning (2017), trained to predict syntactic dependencies. Let  $A_{\text{parse}}$  be the parse attention weights, at layer  $l$ . Its input is the matrix of token representations  $S^{(l-1)}$ . As with the other attention heads, we project  $S^{(l-1)}$  into key, value and query representations, denoted  $K_{\text{parse}}, Q_{\text{parse}}, V_{\text{parse}}$ . Here the key and query projections correspond to *parent* and *dependent* representations of the tokens, and we allow their dimensions to differ from the rest of the attention heads to more closely follow the implementation of Drazot and Manning (2017). Unlike the other attention heads which use a dot product to score key-query pairs, we score the compatibility between  $K_{\text{parse}}$  and  $Q_{\text{parse}}$  using a bi-affine operator  $U_{\text{heads}}$  to obtain attention weights:

$$A_{\text{parse}} = \text{softmax}(Q_{\text{parse}} U_{\text{heads}} K_{\text{parse}}^T) \quad (4)$$

These attention weights are used to compose a weighted average of the value representations  $V_{\text{parse}}$  as in the other attention heads. We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $t$  to a candidate head  $q$  as  $A_{\text{parse}}[t, q]$ , we model the probability of token  $t$  having parent  $q$  as:

$$P(q = \text{head}(t) \mid X) = A_{\text{parse}}[t, q] \quad (5)$$

using the attention weights  $A_{\text{parse}}[t]$  as the distribution over possible heads for token  $t$ . We define the root token as having a self-loop. This attention head thus emits a directed graph<sup>1</sup> where each token's parent is the token to which the attention  $A_{\text{parse}}$  assigns the highest weight.

We also predict dependency labels using per-class bi-affine operations between parent and dependent representations  $Q_{\text{parse}}$  and  $K_{\text{parse}}$  to produce per-label scores, with locally normalized probabilities over dependency labels  $y_t^{(l)}$  given by the softmax function. We refer the reader to Drazot and Manning (2017) for more details.

This attention head now becomes an oracle for syntax, denoted  $\mathcal{P}$ , providing a dependency parse to downstream layers. This model not only predicts its own dependency axes, but allows for the injection of auxiliary parse information at test time by simply setting  $A_{\text{parse}}$  to the parse parents produced by e.g. a state-of-the-art parser. In this way, our model can benefit from improved, external parsing models without re-training. Unlike typical multi-task models, ours maintains the ability to leverage external syntactic information.

### 2.3 Multi-task learning

We also share the parameters of lower layers in our model to predict POS tags and predicates. Following He et al. (2017), we focus on the end-of-sentence setting, where predicates must be predicted on-the-fly. Since we also train our model to predict syntactic dependencies, it is beneficial to give the model knowledge of POS information. While much previous work employs a pipelined approach to both POS tagging for dependency parsing and predicate detection for SRL, we take a multi-task learning (MTL) approach (Caruana,

<sup>1</sup>Usually the head emits a tree, but we do not enforce it here.

	Dev			WSI Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
GloVe	81.8	81.2	81.5	82.0	83.4	82.7	69.7	70.5	70.1
He et al. (2017) PoE	81.3	81.9	81.6	81.2	83.9	82.5	69.7	71.9	70.8
SA	83.52	81.28	82.39	84.17	83.28	83.72	72.98	70.1	71.51
LISA	83.81	81.39	82.24	84.07	83.16	83.61	73.32	70.56	71.91
+D&M	<b>84.59</b>	<b>82.59</b>	<b>83.58</b>	<b>85.53</b>	<b>84.45</b>	<b>84.99</b>	<b>75.8</b>	<b>73.54</b>	<b>74.66</b>
+Gold	87.91	85.73	86.81	—	—	—	—	—	—
ELMo	—	—	—	—	—	—	—	—	—
He et al. (2018)	84.9	<b>85.7</b>	85.3	84.8	<b>87.2</b>	86.0	73.9	<b>78.4</b>	76.1
SA	85.78	84.74	85.26	86.21	85.98	86.09	77.1	75.61	76.35
LISA	<b>86.07</b>	84.64	<b>85.35</b>	86.69	86.42	86.55	78.95	77.17	78.05
+D&M	85.83	84.51	85.17	<b>87.13</b>	<b>86.67</b>	<b>86.90</b>	<b>79.02</b>	77.49	<b>78.25</b>
+Gold	88.51	86.77	87.63	—	—	—	—	—	—

Table 1: Precision, recall and F1 on the CoNLL-2005 development and test sets.

WSI Test	P	R	F1
He et al. (2018)	84.2	83.7	83.9
Tan et al. (2018)	84.5	85.2	84.8
SA	84.7	84.24	84.47
LISA	84.72	84.57	84.64
+D&M	<b>86.02</b>	<b>86.05</b>	<b>86.04</b>
Brown Test	P	R	F1
He et al. (2018)	74.2	73.1	73.7
Tan et al. (2018)	73.5	74.6	74.1
LISA	73.89	72.39	73.13
LISA	74.77	74.32	74.55
+D&M	<b>76.65</b>	<b>76.44</b>	<b>76.54</b>

Table 2: Precision, recall and F1 on CoNLL-2005 with gold predicates.

### 4.1 Semantic role labeling

Table 1 lists precision, recall and F1 on the CoNLL-2005 development and test sets using predicted predicates. For models using GloVe embeddings, our syntax-free SA model already achieves a new state-of-the-art by jointly predicting predicates, POS and SRL. LISA with its own parses performs comparably to SA, but when supplied with D&M parses LISA out-performs the previous state-of-the-art by 2.5 F1 points. On the out-of-domain Brown test set, LISA also performs comparably to its syntax-free counterpart with its own parses, but with D&M parses LISA performs exceptionally well, more than 3.5 F1 points higher than He et al. (2018). Incorporating ELMo em-

beddings improves all scores. The gap in SRL F1 between models using LISA and D&M parses is smaller due to LISA's improved parsing accuracy (see §4.2), but LISA with D&M parses still achieves the highest F1: nearly 1.0 absolute F1 higher than the previous state-of-the-art on WSI, and more than 2.0 F1 higher on Brown. In both settings LISA leverages domain-agnostic syntactic information rather than over-fitting to the newswire training data, which leads to high performance even on out-of-domain test.

To compare to more prior work we also evaluate our models in the artificial setting where gold predicates are provided at test time. For fair comparison we use GloVe embeddings, provide predicate identifier embeddings on the input and re-encode the sequence relative to each gold predicate. Here LISA still excels: with D&M parses, LISA out-performs the previous state-of-the-art by more than 2 F1 on both WSI and Brown.

Table 3 reports precision, recall and F1 on the CoNLL-2012 test set. We observe performance similar to that observed on CoNLL-2005: Using GloVe embeddings our SA baseline already out-performs He et al. (2018) by nearly 1.5 F1. With its own parses, LISA slightly under-performs our syntax-free model, but when provided with stronger D&M parses LISA out-performs the state-of-the-art by more than 2.5 F1. Like CoNLL-2005, ELMo representations improve all models and close the F1 gap between models supplied with LISA and D&M parses. On this dataset ELMo also substantially narrows the

# acronyms

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>2</sup>, David Weiss<sup>2</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences  
University of Massachusetts Amherst  
{strubell, pat, mcallum}@cs.umass.edu  
<sup>2</sup>Google AI Language  
New York, NY  
{andor, djweiss}@google.com

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA): a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and SRL. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA incorporates syntax using merely raw tokens as input, encoding the sequence of tokens to simultaneously perform parsing, predicate detection and role labeling for all predicted syntactic heads. LISA is trained by training on a high-quality ready available, if not a high-quality, over time without the need for manual annotation. In experiments, LISA achieves new state-of-the-art results on a standard word embeddings, attaining 2.5 F1 absolute higher than the previous state-of-the-art on newswire and more than 3.5 F1 on out-of-domain data, nearly 10% reduction in error. On CoNLL-2012 English SRL we also show an improvement of more than 2.5 F1. LISA also out-performs the state-of-the-art with contextually-encoded (ELMo) word representations, by nearly 1.0 F1 on news and more than 2.0 F1 on out-of-domain test.

### 1 Introduction

Semantic role labeling (SRL) extracts a high-level representation of meaning from a sentence, labeling e.g. who did what to whom. Explicit representations of such semantic information have been

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazzafshan and Gildea, 2013). Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Panyanov et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models. Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017) indicate that many of the errors made by a syntax-free neural network on SRL are tied to certain syntactic confusions such as prepositional phrase attachment, and show that while constrained inference using a relatively low-accuracy predicted parse can provide small improvements in SRL accuracy, providing a gold-quality parse leads to substantial gains. Marcheggiani and Titov (2017) incorporate syntax from a high-quality parser (Kiparwasser and Goldberg, 2016) using graph convolutional neural networks (Kipf and Welling, 2017), but like He et al. (2017) they attain only small increases over a model with no syntactic parse, and even perform worse than a syntax-free model on out-of-domain data. These works suggest that though syntax has the potential to improve neural network SRL models, we have not

and normalize with the softmax function to produce a distinct distribution of the value representations  $V_{parent}$  as in the other attention heads.

We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $t$  to a candidate head  $q$  as  $A_{parent}[t, q]$ , we model the probability of token  $t$  having parent  $q$  as:

$$P(q = \text{head}(t) \mid X) = A_{parent}[t, q] \quad (5)$$

using the attention weights  $A_{parent}[t]$  as the distribution over possible heads for token  $t$ . We define the root token as having a self-loop. This attention head thus emits a directed graph<sup>1</sup> where each token's parent is the token to which the attention  $A_{parent}$  assigns the highest weight.

We also predict dependency labels using per-class bi-affine operations between parent and dependent representations  $Q_{parent}$  and  $K_{parent}$  to produce per-label scores, with locally normalized probabilities over dependency labels  $y$  given by the softmax function. We refer the reader to Dozat and Manning (2017) for more details.

### 2.2 Syntactically-informed self-attention

Typically, neural attention mechanisms are left on their own to learn to attend to relevant inputs. Instead, we propose training the self-attention to attend to specific tokens corresponding to the syntactic structure of the sentence as a mechanism for passing linguistic knowledge to later layers.

Specifically, we replace one attention head with the deep bi-affine model of Dozat and Manning (2017), trained to predict syntactic dependencies. Let  $A_{parent}$  be the parse attention weights, at layer  $l$ . Its input is the matrix of token representations  $S^{(l-1)}$ . As with the other attention heads, we project  $S^{(l-1)}$  into key, value and query representations, denoted  $K_{parent}$ ,  $Q_{parent}$ ,  $V_{parent}$ . Here the key and query projections correspond to *parent* and *dependent* representations of the tokens, and we allow their dimensions to differ from the rest of the attention heads to more closely follow the implementation of Dozat and Manning (2017). Unlike the other attention heads which use a dot product to score key-query pairs, we score the compatibility between  $K_{parent}$  and  $Q_{parent}$  using a bi-affine operator  $U_{heads}$  to obtain attention weights:

$$A_{parent} = \text{softmax}(Q_{parent} U_{heads} K_{parent}^T) \quad (4)$$

These attention weights are used to compose a weighted average of the value representations  $V_{parent}$  as in the other attention heads. We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $t$  to a candidate head  $q$  as  $A_{parent}[t, q]$ , we model the probability of token  $t$  having parent  $q$  as:

$$P(q = \text{head}(t) \mid X) = A_{parent}[t, q] \quad (5)$$

using the attention weights  $A_{parent}[t]$  as the distribution over possible heads for token  $t$ . We define the root token as having a self-loop. This attention head thus emits a directed graph<sup>1</sup> where each token's parent is the token to which the attention  $A_{parent}$  assigns the highest weight.

We also predict dependency labels using per-class bi-affine operations between parent and dependent representations  $Q_{parent}$  and  $K_{parent}$  to produce per-label scores, with locally normalized probabilities over dependency labels  $y$  given by the softmax function. We refer the reader to Dozat and Manning (2017) for more details.

This attention head now becomes an oracle for syntax, denoted  $P$ , providing a dependency parse to downstream layers. This model not only predicts its own dependency axes, but allows for the injection of auxiliary parse information at test time by simply setting  $A_{parent}$  to the parse parents provided by e.g. a state-of-the-art parser. In this way, our model can benefit from improved, external parsing models without re-training. Unlike typical multi-task models, ours maintains the ability to leverage external syntactic information.

### 2.3 Multi-task learning

We also share the parameters of lower layers in our model to predict POS tags and predicates. Following He et al. (2017), we focus on the end-of-sentence setting, where predicates must be predicted on-the-fly. Since we also train our model to predict syntactic dependencies, it is beneficial to give the model knowledge of POS information. While much previous work employs a pipelined approach to both POS tagging for dependency parsing and predicate detection for SRL, we take a multi-task learning (MTL) approach (Caruana,

<sup>1</sup>Usually the head emits a tree, but we do not enforce it here.

GloVe	Dev			WSI Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2017) PoE	81.8	81.2	81.5	82.0	83.4	82.7	69.7	70.5	70.1
He et al. (2018)	81.3	81.9	81.6	81.2	83.9	82.5	69.7	71.9	70.8
LISA	83.52	81.28	82.39	84.17	83.28	83.72	72.98	70.1	71.51
+D&M	83.1	81.39	82.24	84.07	83.16	83.61	73.32	70.56	71.91
+Gold	84.20	82.59	83.58	85.53	84.45	84.99	75.8	73.54	74.66
+D&M	85.73	86.81	86.27	—	—	—	—	—	—

Table 1: Precision, recall and F1 on the CoNLL-2005 development and test sets.

WSI Test	P	R	F1
	P	R	F1
He et al. (2018)	84.2	83.7	83.9
Tan et al. (2018)	84.5	85.2	84.8
SA	84.7	84.24	84.47
LISA	84.72	84.57	84.64
+D&M	86.02	86.05	86.04

Table 2: Precision, recall and F1 on CoNLL-2005 with gold predicates.

### 4.1 Semantic role labeling

Table 1 lists precision, recall and F1 on the CoNLL-2005 development and test sets using predicted predicates. For models using GloVe embeddings, our syntax-free SA model already achieves a new state-of-the-art by jointly predicting predicates, POS and SRL. LISA with its own parses performs comparably to SA, but when supplied with D&M parses LISA outperforms the previous state-of-the-art by 2.5 F1 points. On the out-of-domain Brown test set, LISA also performs comparably to its syntax-free counterpart with its own parses, but with D&M parses LISA performs exceptionally well, more than 3.5 F1 points higher than He et al. (2018). Incorporating ELMo em-

beddings improves all scores. The gap in SRL F1 between models using LISA and D&M parses is smaller due to LISA's improved parsing accuracy (see §4.2), but LISA with D&M parses still achieves the highest F1, nearly 1.0 absolute F1 higher than the previous state-of-the-art on WSI, and more than 2.0 F1 higher on Brown. In settings LISA leverages domain-agnostic syntactic information rather than over-fitting to the training data which leads to high performance on out-of-domain test. To compare to more prior work we also evaluate models in the artificial setting where gold parses are provided at test time. For fair comparison, we use GloVe embeddings, provide predicate indicator embeddings on the input and re-encode the sequence relative to each gold predicate. Here LISA still excels: with D&M parses, LISA outperforms the previous state-of-the-art by more than 2 F1 on both WSI and Brown.

Table 3 reports precision, recall and F1 on the CoNLL-2012 test set. We observe performance similar to that observed on CoNLL-2005: Using GloVe embeddings our SA baseline already outperforms He et al. (2018) by nearly 1.5 F1. With its own parses, LISA slightly underperforms our syntax-free model, but when provided with stronger D&M parses LISA outperforms the state-of-the-art by more than 2.5 F1. Like CoNLL-2005, ELMo representations improve all models and close the F1 gap between models supplied with LISA and D&M parses. On this dataset ELMo also substantially narrows the

# acronyms • symbols

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>1</sup>, David Weiss<sup>1</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences  
University of Massachusetts Amherst  
{strubell, pat, mcallum}@ics.umass.edu  
{goolge, djweiss}@google.com  
New York, NY

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA): a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and SRL. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA incorporates syntax using merely raw tokens as input, encoding the sequence of tokens to simultaneously perform parsing, predicate detection and role labeling for all predicted syntactic heads. LISA is trained by training on a multi-task dataset, where a high-quality ready available, it is available at test time without incurring additional costs. In experiments, LISA achieves new state-of-the-art on out-of-domain data, nearly 10% reduction in error. On CoNLL-2012 English SRL, we also show an improvement of more than 2.5 F1. LISA also outperforms the state-of-the-art with contextually-encoded (ELMo) word representations, by nearly 1.0 F1 on news and more than 2.0 F1 on out-of-domain test.

### 1 Introduction

Semantic role labeling (SRL) extracts a high-level representation of meaning from a sentence, labeling e.g. who did what to whom. Explicit representations of such semantic information have been

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berrant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazzafshan and Gildea, 2013). Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Panyanov et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models.

Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017) indicate that many of the errors made by a syntax-free neural model on SRL are tied to certain syntactic confusions such as prepositional phrase attachment, and show that while constrained inference using a relatively low-accuracy predicted parse can provide small improvements in SRL accuracy, providing a gold-quality parse leads to substantial gains. Marcheggiani and Titov (2017) incorporate syntax from a high-quality parser (Kiparwasser and Goldberg, 2016) using graph convolutional neural networks (Kipf and Welling, 2017), but like He et al. (2017) they attain only small improvements over a model with no syntactic parse, and even perform worse than a syntax-free model on out-of-domain data. These works suggest that though syntax has the potential to improve neural network SRL models, we have not

and normalize with the softmax function to produce a distinct distribution for each token over all the tokens in the sentence:

$$A_i^{(l)} = \text{softmax}(d_i^{(l)} \odot q_i^{(l)} K_i^{(l)T}) \quad (2)$$

These attention weights are then multiplied by  $V_i^{(l)}$  for each token to obtain the self-attended token representations  $M_i^{(l)}$ :

$$M_i^{(l)} = A_i^{(l)} V_i^{(l)} \quad (3)$$

Row  $i$  of  $M^{(l)}$ , the self-attended representation for token  $i$  at layer  $l$ , is thus the weighted sum of all token representations in  $V_i^{(l)}$  with weights given by  $A_i^{(l)}$ .

The outputs of all attention heads are concatenated, and this is repeated for the feed-forward layer, which linear projections each followed by activations (Maas et al., 2013). The output of the feed-forward to the initial representation and apply layer normalization to give the final output of self-attention layer  $j$ , as in Eqn. 1.

### 2.2 Syntactically-informed self-attention

Typically, neural attention mechanisms are left on their own to learn to attend to relevant inputs. Instead, we propose training the self-attention to attend to specific tokens corresponding to the syntactic structure of the sentence as a mechanism for passing linguistic knowledge to later layers.

Specifically, we replace one attention head with the deep bi-affine model of Dozat and Manning (2017), trained to predict syntactic dependencies. Let  $A_{\text{parse}}$  be the parse attention weights, at layer  $l$ . Its input is the matrix of token representations  $S^{(l-1)}$ . As with the other attention heads, we project  $S^{(l-1)}$  into key, value and query representations, denoted  $K_{\text{parse}}, Q_{\text{parse}},$  and  $V_{\text{parse}}$ . Here the key and query projections correspond to *parent* and *dependent* representations of the tokens, and we allow their dimensions to differ from the rest of the attention heads to more closely follow the implementation of Dozat and Manning (2017). Unlike the other attention heads which use a dot product to score key-query pairs, we score the compatibility between  $K_{\text{parse}}$  and  $Q_{\text{parse}}$  using a bi-affine operator  $U_{\text{heads}}$  to obtain attention weights:

$$A_{\text{parse}} = \text{softmax}(Q_{\text{parse}} U_{\text{heads}} K_{\text{parse}}^T) \quad (4)$$

These attention weights are used to compose a weighted average of the value representations  $V_{\text{parse}}$  as in the other attention heads.

We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $t$  to a candidate head  $q$  as  $A_{\text{parse}}[t, q]$ , we model the probability of token  $t$  having parent  $q$  as:

$$P(q = \text{head}(t) \mid X) = A_{\text{parse}}[t, q] \quad (5)$$

The attention weights  $A_{\text{parse}}[t]$  as the distribution over possible heads for token  $t$ . We define each token as having a self-loop. This attention thus emits a directed graph<sup>1</sup> where each parent is the token to which the attention assigns the highest weight.

We also predict dependency labels using per-affine operations between parent and dependent representations  $Q_{\text{parse}}$  and  $K_{\text{parse}}$  to produce per-label scores, with locally normalized probabilities over dependency labels  $y_i^l$  given by the softmax function. We refer the reader to Dozat and Manning (2017) for more details.

This attention head now becomes an oracle for syntax, denoted  $P$ , providing a dependency parse downstream layers. This model not only predicts its own dependency axes, but allows for the injection of auxiliary parse information at test time by simply setting  $A_{\text{parse}}$  to the parse parents produced by e.g. a state-of-the-art parser. In this way, our model can benefit from improved, external parsing models without incurring additional multi-task models, or to leverage external syntax.

### 2.3 Multi-task learning

We also share the parameter model to predict POS following He et al. (2017), and setting, where predicates must be predicted on-the-fly. Since we also train our model to predict syntactic dependencies, it is beneficial to give the model knowledge of POS information. While much previous work employs a pipelined approach to both POS tagging for dependency parsing and predicate detection for SRL, we take a multi-task learning (MTL) approach (Caruana,

<sup>1</sup>Usually the head emits a tree, but we do not enforce it here.

GloVe	Dev			WSI Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2017) PoE	81.8	81.2	81.5	82.0	83.4	82.7	69.7	70.5	70.1
He et al. (2018)	81.3	81.9	81.6	81.2	83.9	82.5	69.7	71.9	70.8
LISA	83.52	81.28	82.39	84.17	83.28	83.72	72.98	70.1	71.51
+D&M	83.1	81.39	82.24	84.07	83.16	83.61	73.32	70.56	71.91
+Gold	84.70	82.59	83.58	85.53	84.45	84.99	75.8	73.54	74.66
+D&M+Gold	85.73	86.81	86.27	—	—	—	—	—	—
ELMo	Dev			WSI Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2018)	85.7	85.3	84.8	87.2	86.0	86.6	73.9	78.4	76.1
SA	84.74	85.26	86.21	85.98	86.09	86.03	77.1	75.61	76.35
LISA	84.64	85.35	86.69	86.42	86.55	86.48	78.95	77.17	78.05
+D&M	84.51	85.17	87.13	86.67	86.90	86.78	79.02	77.49	78.25
+Gold	88.51	86.77	87.63	—	—	—	—	—	—

Table 1: Precision, recall and F1 on the CoNLL-2005 development and test sets.

WSI Test	P	R	F1
	P	R	F1
He et al. (2018)	84.2	83.7	83.9
Tan et al. (2018)	84.5	85.2	84.8
SA	84.7	84.24	84.47
LISA	84.72	84.57	84.64
+D&M	86.02	86.05	86.04
Brown Test	P	R	F1
	P	R	F1
He et al. (2018)	73.9	78.4	76.1
Tan et al. (2018)	77.1	75.61	76.35
LISA	78.95	77.17	78.05
+D&M	79.02	77.49	78.25

Table 2: Precision, recall and F1 on CoNLL-2005 with gold predicates.

### 4.1 Semantic role labeling

Table 1 lists precision, recall and F1 on the CoNLL-2005 development and test sets using predicted predicates. For models using GloVe embeddings, our syntax-free SA model already achieves a new state-of-the-art by jointly predicting predicates, POS and SRL. LISA with its own parses performs comparably to SA, but when supplied with D&M parses LISA outperforms the previous state-of-the-art by 2.5 F1 points. On the out-of-domain Brown test set, LISA also performs comparably to its syntax-free counterpart with its own parses, but with D&M parses LISA performs exceptionally well, more than 3.5 F1 points higher than He et al. (2018). Incorporating ELMo em-

beddings improves all scores. The gap in SRL F1 between models using LISA and D&M parses is smaller due to LISA's improved parsing accuracy (see §4.2), but LISA with D&M parses still achieves the highest F1: nearly 1.0 absolute F1 higher than the previous state-of-the-art on WSI, and more than 2.0 F1 higher on Brown.

Settings LISA leverages domain-agnostic information rather than over-fitting to the training data which leads to high performance on out-of-domain test.

Compared to prior work we also evaluate models in the artificial setting where gold parses are provided at test time. For fair comparison, we use GloVe embeddings, provide predicate indicator embeddings on the input and re-encode the sequence relative to each gold predicate. Here LISA still excels: with D&M parses, LISA outperforms the previous state-of-the-art by more than 2 F1 on both WSI and Brown.

Table 3 reports precision, recall and F1 on the CoNLL-2012 test set. We observe performance similar to that observed on CoNLL-2005: Using GloVe embeddings our SA baseline already outperforms He et al. (2018) by nearly 1.5 F1. With its own parses, LISA slightly underperforms our syntax-free model, but when provided with stronger D&M parses LISA outperforms the state-of-the-art by more than 2.5 F1. Like CoNLL-2005, ELMo representations improve all models and close the F1 gap between models supplied with LISA and D&M parses. On this dataset ELMo also substantially narrows the





# acronyms • symbols • new terms

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>2</sup>, David Weiss<sup>2</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences

University of Massachusetts Amherst

{strubell, pat, mcallum}@cs.umass.edu

<sup>2</sup>Google AI Language

New York, NY

{andor, djweiss}@google.com

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA), a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and NER. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA can incorporate syntax using merely raw tokens as input, encoding the sequence only once to simultaneously perform parsing, predicate detection and role labeling for all predicates. Syntax is incorporated by training one attention head to attend to syntactic parents for each token. Moreover, if a high-quality syntactic parse is already available, it can be beneficially injected at test time without re-training our LISA model. In experiments on [Penn Treebank](#) and [LISA](#), LISA achieves new state-of-the-art performance for a model using predicted predicates and standard word embeddings, attaining 2.5 F1 absolute higher than the previous state-of-the-art on newswire and more than 3.5 F1 on out-of-domain data, nearly 10% reduction in error. On [LISA](#) and [LISA](#) datasets, we also show an improvement of more than 2.5 F1. LISA also out-performs the state-of-the-art with contextually-encoded (ELMo) word representations, by nearly 1.0 F1 on news and more than 2.0 F1 on out-of-domain test.

### 1 Introduction

Semantic role labeling (SRL) extracts a high-level representation of meaning from a sentence, labeling e.g. who did what to whom. Explicit representations of such semantic information have been

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazzafshan and Gildea, 2013). Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Panyanov et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models.

Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017) indicate that many of the errors made by a syntax-free neural network on SRL are tied to certain syntactic confusions such as prepositional phrase attachment, and show that while constrained inference using a relatively low-accuracy predicted parse can provide small improvements in LISA accuracy, providing a gold-quality parse leads to substantial gains. Marcheggiani and Titov (2017) incorporate syntax from a high-quality parser (Kiparwasser and Goldberg, 2016) using graph convolutional neural networks (Kipf and Welling, 2017), but like He et al. (2017) they attain only small increases over a model with no syntactic parse, and even perform worse than a syntax-free model on out-of-domain data. These works suggest that though syntax has the potential to improve neural network SRL models, we have not

and normalize with the softmax function to produce a distinct distribution for each token over all the tokens in the sentence:

$$p_i = \text{softmax}_j \left( \frac{v_i \cdot w_j}{\sum_k v_i \cdot w_k} \right) \quad (2)$$

These attention weights are then multiplied by  $v_i$  for each token to obtain the self-attended token representations  $\tilde{v}_i$ :

$$\tilde{v}_i = p_i \cdot v_i \quad (3)$$

Row  $\tilde{v}_i$  is the self-attended representation for token  $i$ . Layer  $\tilde{v}_i$  is thus the weighted sum with respect to  $v_i$  (with weights given by  $p_i$ ) over the token representations in  $v_i$ .

The outputs of all attention heads for each token are concatenated, and this representation is passed to the feed-forward layer, which consists of two linear projections each followed by leaky ReLU activations (Maas et al., 2013). We add the output of the feed-forward to the initial representation and apply layer normalization to give the final output of self-attention layer  $\tilde{v}_i$  as in Eq. 1.

### 2.2 Syntactically-informed self-attention

Typically, neural attention mechanisms are left on their own to learn to attend to relevant inputs. Instead, we propose training the self-attention to attend to specific tokens corresponding to the syntactic structure of the sentence as a mechanism for passing linguistic knowledge to later layers.

Specifically, we replace one attention head with the deep bi-affine model of Draz and Manning (2017), trained to predict syntactic dependencies. Let  $\tilde{v}_i$  be the parse attention weights, at layer  $i$  its input is the matrix of token representations  $v_i$ . As with the other attention heads, we project  $\tilde{v}_i$  into key, value and query representations, denoted  $\tilde{v}_i^k$ ,  $\tilde{v}_i^v$  and  $\tilde{v}_i^q$ . Here the key and query projections correspond to  $\tilde{v}_i^k$  and  $\tilde{v}_i^q$  and the representations of the tokens, and we allow their dimensions to differ from the rest of the attention heads to more closely follow the implementation of Draz and Manning (2017). Unlike the other attention heads which use a dot product to score key-query pairs, we score the compatibility between  $\tilde{v}_i^k$  and  $\tilde{v}_i^q$  using a bi-affine operator  $U_{\text{LISA}}$  to obtain attention weights:

$$\tilde{v}_i = \text{softmax}_j \left( \frac{U_{\text{LISA}}(\tilde{v}_i^k, \tilde{v}_i^q)}{\sum_k U_{\text{LISA}}(\tilde{v}_i^k, \tilde{v}_i^q)} \right) \quad (4)$$

These attention weights are used to compose a weighted average of the value representations  $\tilde{v}_i^v$  in the other attention heads:

We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $i$  to a candidate head  $j$  as  $\tilde{v}_i^j$ , we model the probability of token  $i$  having parent  $j$  as:

$$p_{ij} = \text{head}_j \left( \frac{\tilde{v}_i^j \cdot \tilde{v}_i^j}{\sum_k \tilde{v}_i^k \cdot \tilde{v}_i^k} \right) \quad (5)$$

using the attention weights  $\tilde{v}_i^j$  as the distribution over possible heads for token  $i$ . We define the root token as having a self-loop. This attention head thus emits a directed graph<sup>1</sup> where each token's parent is the token to which the attention is assigned the highest weight.

We also predict dependency labels using per-class bi-affine operations between parent and dependent representations  $\tilde{v}_i^k$  and  $\tilde{v}_i^q$  to produce per-label scores, with locally normalized probabilities over dependency labels  $\tilde{v}_i^l$  given by the softmax function. We refer the reader to Draz and Manning (2017) for more details.

This attention head now becomes an oracle for syntax, denoted  $\tilde{v}_i^s$  providing a dependency parse to downstream layers. This model not only predicts its own dependency axes, but allows for the injection of auxiliary parse information at test time by simply setting  $\tilde{v}_i^s$  to the parse parents produced by e.g. a state-of-the-art parser. In this way, our model can benefit from improved, external parsing models without re-training. Unlike typical multi-task models, ours maintains the ability to leverage external syntactic information.

### 2.3 Multi-task learning

We also share the parameters of lower layers in our model to predict [SRL](#) tags and predicates. Following He et al. (2017), we focus on the end-to-end setting, where predicates must be predicted on-the-fly. Since we also train our model to predict syntactic dependencies, it is beneficial to give the model knowledge of [SRL](#) information. While much previous work employs a pipelined approach to both [SRL](#) tagging for dependency parsing and predicate detection for [SRL](#), we take a multi-task learning [SRL](#) approach (Caruana,

<sup>1</sup>Usually the head emits a tree, but we do not enforce it here.

GloVe	Dev			LISA Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2017)	81.8	81.2	81.5	82.0	81.4	81.7	69.7	70.1	70.1
He et al. (2018)	81.3	81.9	81.6	81.2	83.9	82.5	69.7	71.9	70.8
LISA	83.52	81.28	82.39	84.17	83.28	83.72	72.98	70.1	71.51
LISA	83.1	81.39	82.24	84.07	83.16	83.61	73.32	70.56	71.91
LISA	84.59	82.59	83.58	85.53	84.45	84.99	75.8	73.54	74.66
LISA	87.91	85.73	86.81	—	—	—	—	—	—

ELMo	Dev			LISA Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2018)	84.9	85.7	85.3	84.8	87.2	86.0	73.9	78.4	76.1
LISA	85.78	84.74	85.26	86.21	85.98	86.09	77.1	75.61	76.35
LISA	86.07	84.64	85.35	86.69	86.42	86.55	78.95	77.17	78.05
LISA	85.83	84.51	85.17	87.13	86.67	86.90	79.02	77.49	78.25
LISA	88.51	86.77	87.63	—	—	—	—	—	—

Table 1: Precision, recall and F1 on the [SRL](#) development and test sets.

LISA Test	P			F1		
	P	R	F1	P	R	F1
He et al. (2018)	84.2	83.7	83.9	—	—	—
Tan et al. (2018)	84.5	85.2	84.8	—	—	—
LISA	84.7	84.24	84.47	—	—	—
LISA	84.72	84.57	84.64	—	—	—
LISA	86.02	86.05	86.04	—	—	—

Brown Test	P			F1		
	P	R	F1	P	R	F1
He et al. (2018)	74.2	73.1	73.7	—	—	—
Tan et al. (2018)	73.5	74.6	74.1	—	—	—
LISA	73.89	72.39	73.13	—	—	—
LISA	74.77	74.32	74.55	—	—	—
LISA	76.65	76.44	76.54	—	—	—

Table 2: Precision, recall and F1 on the [SRL](#) development and test sets with gold predicates.

### 4.1 Semantic role labeling

Table 1 lists precision, recall and F1 on the [SRL](#) development and test sets using predicted predicates. For models using GloVe embeddings, our syntax-free [SRL](#) model already achieves a new state-of-the-art by jointly predicting predicate [SRL](#) and [SRL](#) with its own parse performs comparably to [SRL](#), but when supplied with [SRL](#) parses [SRL](#) out-performs the previous state-of-the-art by 2.5 F1 points. On the out-of-domain Brown test set [SRL](#) also performs comparably to its syntax-free counterpart with its own parses, but with [SRL](#) parses [SRL](#) performs exceptionally well, more than 3.5 F1 points higher than He et al. (2018). Incorporating ELMo em-

beddings improves all scores. The gap to [SRL](#) F1 between models using [SRL](#) and [SRL](#) parses is smaller due to [SRL](#)'s improved parsing accuracy (see §4.2), but [SRL](#) with [SRL](#) parses still achieves the highest F1, nearly 1.0 absolute F1 higher than the previous state-of-the-art on [SRL](#), and more than 2.0 F1 higher on Brown. In both settings [SRL](#) leverages domain-agnostic syntactic information rather than over-fitting to the newswire training data, which leads to high performance even on out-of-domain test.

To compare to more prior work we also evaluate our models in the artificial setting where gold predicates are provided at test time. For fair comparison we use GloVe embeddings, provide predicate indicator embeddings on the input and re-encode the sequence relative to each gold predicate. Here [SRL](#) still excels: with [SRL](#) parses, [SRL](#) out-performs the previous state-of-the-art by more than 2 F1 on both [SRL](#) and Brown.

Table 3 reports precision, recall and F1 on the [SRL](#) test set. We observe performance similar to that observed on [SRL](#). Using GloVe embeddings our [SRL](#) baseline already out-performs He et al. (2018) by nearly 1.5 F1. With its own parses, [SRL](#) slightly under-performs our syntax-free model, but when provided with stronger [SRL](#) parses [SRL](#) out-performs the state-of-the-art by more than 2.5 F1. Like [SRL](#) and [SRL](#), ELMo representations improve all models and close the F1 gap between models supplied with [SRL](#) and [SRL](#) parses. On this dataset ELMo also substantially narrows the

# ScholarPhi

an augmented reading interface that  
surfaces just-in-time, position-sensitive  
definitions of nonce words.

See demo video at <https://youtu.be/yYcQf-Yq8B0>

# ScholarPhi

an augmented reading interface that  
surfaces just-in-time, position-sensitive  
definitions of nonce words.

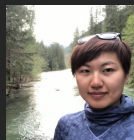
# SIMPLIFYING THE COMPLEX

Paper Plain: Making Medical Research Papers Approachable to  
Healthcare Consumers with Natural Language Processing

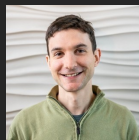
@ACM TOCHI '23.



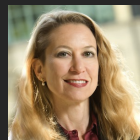
Tal  
August



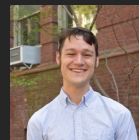
Lucy Lu Wang



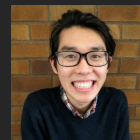
Jonathan Bragg



Marti A. Hearst



Andrew Head



Kyle Lo

# Complex Content: How to Understand it?

EXPERT OPINION ON INVESTIGATIONAL DRUGS  
2020, VOL. 29, NO. 8, 845–867  
<https://doi.org/10.1080/13543784.2020.1777983>



## REVIEW

## Therapeutic peptides for the treatment of systemic lupus erythematosus: a place in therapy

Rossella Talotta<sup>a</sup>, Fabiola Atzeni<sup>a</sup> and Magdalena Janina Laska<sup>b</sup>

<sup>a</sup>Department of Clinical and Experimental Medicine, Rheumatology Unit, Azienda Ospedaliera "Gaetano Martino", University of Messina, Messina, Italy; <sup>b</sup>Department of Rheumatology, Aarhus University Hospital, Aarhus, Denmark

### ABSTRACT

**Introduction:** Studies *in vitro* and *in vivo* have identified several peptides that are potentially useful in treating systemic lupus erythematosus (SLE). The rationale for their use lies in the cost-effective production, high potency, target selectivity, low toxicity, and a peculiar mechanism of action that is mainly based on the induction of immune tolerance. Three therapeutic peptides have entered clinical development, but they have yielded disappointing results. However, some subsets of patients, such as those with the positivity of anti-dsDNA antibodies, appear more likely to respond to these medications.

**Areas covered:** This review evaluates the potential use of therapeutic peptides for SLE and gives an opinion on how they may offer advantages for SLE treatment.

**Expert opinion:** Given their acceptable safety profile, therapeutic peptides could be added to agents traditionally used to treat SLE and this may offer a synergistic and drug-sparing effect, especially in selected patient populations. Moreover, they could temporarily be utilized to manage SLE flares, or be administered as a vaccine in subjects at risk. Efforts to ameliorate bioavailability, increase the half-life and prevent immunogenicity are ongoing. The formulation of hybrid compounds, like peptibodies or peptidomimetic small molecules, is expected to yield renewed treatments with a better pharmacologic profile and increased efficacy.

### ARTICLE HISTORY

Received 23 January 2020  
Accepted 1 June 2020

### KEYWORDS

Pathogenesis; systemic lupus erythematosus; therapeutic peptides; treatment; peptibodies; peptides

## 1. Introduction

Systemic lupus erythematosus (SLE) is the prototypical autoimmune connective tissue disease, affecting 5 million individuals worldwide, mainly women during the fertile age [1]. Clinical presentation broadly varies from patient to patient, with kidney and central nervous system (CNS) involvement representing the most severe complications [2]. The disease

and usually shows less successful results, *Figure 2* [8]. Several immunologic pathways are, in fact, concomitantly activated in SLE, and this justifies the use of medications like steroids, immunosuppressants and disease-modifying anti-rheumatic drugs (DMARDs), which unselectively counteract the immune response. Such a combo-therapy can indeed have many summing side effects that can be further

To help you read this paper, below is a list of key questions that are answered in this paper.

What condition does this paper study?

Systemic Lupus Erythematosus is a disease that affects about 5 million people in the world, mostly women.

see paragraph on page 2

How is the condition usually treated?

After you get the diagnosis of lupus, the doctor will see how bad your lupus is and how much it affects your body.

see paragraph on page 5

What did the paper want to find out?

This is a review of the evidence and reasons why doctors are using peptides to treat lupus.

see paragraph on page 3

What did the paper do?

The next paragraphs tell us about some drugs that are being tested to see if they can help people with lupus.

see paragraph on page 5

EXPERT OPINION ON INVESTIGATIONAL DRUGS  
2020, VOL. 29, NO. 8, 845–867  
<https://doi.org/10.1080/13543784.2020.1777983>



Check for updates

REVIEW

## Therapeutic peptides for the treatment of systemic lupus erythematosus: a place in therapy

Rossella Talotta<sup>a</sup>, Fabiola Atzeni<sup>a</sup> and Magdalena Janina Laska<sup>b</sup>

<sup>a</sup>Department of Clinical and Experimental Medicine, Rheumatology Unit, Azienda Ospedaliera “Gaetano Martino”, University of Messina, Messina, Italy; <sup>b</sup>Department of Rheumatology, Aarhus University Hospital, Aarhus, Denmark

### ABSTRACT

**Introduction:** Studies *in vitro* and *in vivo* have identified several peptides that are potentially useful in treating systemic lupus erythematosus (SLE). The rationale for their use lies in the cost-effective production, high potency, target selectivity, low toxicity, and a peculiar mechanism of action that is mainly based on the induction of immune tolerance. Three therapeutic peptides have entered clinical development, but they have yielded disappointing results. However, some subsets of patients, such as those with the positivity of anti-dsDNA antibodies, appear more likely to respond to these medications.

**Areas covered:** This review evaluates the potential use of therapeutic peptides for SLE and gives an opinion on how they may offer advantages for SLE treatment.

**Expert opinion:** Given their acceptable safety profile, therapeutic peptides could be added to agents traditionally used to treat SLE and this may offer a synergistic and drug-sparing effect, especially in selected patient populations. Moreover, they could temporarily be utilized to manage SLE flares, or be administered as a vaccine in subjects at risk. Efforts to ameliorate bioavailability, increase the half-life and prevent immunogenicity are ongoing. The formulation of hybrid compounds, like peptibodies or peptidomimetic small molecules, is expected to yield renewed treatments with a better pharmacologic profile and increased efficacy.

### ARTICLE HISTORY

Received 23 January 2020  
Accepted 1 June 2020

### KEYWORDS

Pathogenesis; systemic lupus erythematosus; therapeutic peptides; treatment; peptibodies; peptides

### 1. Introduction

Systemic lupus erythematosus (SLE) is the prototypical autoimmune connective tissue disease, affecting 5 million individuals worldwide, mainly women during the fertile age [1]. Clinical presentation broadly varies from patient to patient, with kidney and central nervous system (CNS) involvement representing the most severe complications [2]. The disease

and usually shows less successful results, *Figure 2* [8]. Several immunologic pathways are, in fact, concomitantly activated in SLE, and this justifies the use of medications like steroids, immunosuppressants and disease-modifying anti-rheumatic drugs (DMARDs), which unselectively counteract the immune response. Such a combo-therapy can indeed have many summing side effects that can be further



# Idea: Add Guiding Questions

To help you read this paper, below is a list of key questions that are answered in this paper.

What condition does this paper study?

Systemic Lupus Erythematosus is a disease that affects about 5 million people in the world, mostly women.

see paragraph on [page 2](#)



How is the condition usually treated?

After you get the diagnosis of lupus, the doctor will see how bad your lupus is and how much it affects your body.

see paragraph on [page 5](#)

EXPERT OPINION ON INVESTIGATIONAL DRUGS  
2020, VOL. 29, NO. 8, 845–867  
<https://doi.org/10.1080/13543784.2020.1777983>

## REVIEW

## Therapeutic peptides for the treatment of systemic lupus erythematosus: a review

Rossella Talotta<sup>a</sup>, Fabiola Atzeni<sup>a</sup> and Magdalena Janina Laska<sup>b</sup>

<sup>a</sup>Department of Clinical and Experimental Medicine, Rheumatology Unit, Azienda Ospedaliera di Padova, Padova, Italy; <sup>b</sup>Department of Rheumatology, Aarhus University Hospital, Aarhus, Denmark

### ABSTRACT

**Introduction:** Studies *in vitro* and *in vivo* have identified several peptides that may be used for treating systemic lupus erythematosus (SLE). The rationale for their use is based on their production, high potency, target selectivity, low toxicity, and a peculiar mechanism of action, mainly based on the induction of immune tolerance. Three therapeutic peptides have been developed, but they have yielded disappointing results. However, some studies suggest that those with the positivity of anti-dsDNA antibodies, appear more likely to respond to treatment.

**Areas covered:** This review evaluates the potential use of therapeutic peptides in the treatment of SLE, based on opinion on how they may offer advantages for SLE treatment.

**Expert opinion:** Given their acceptable safety profile, therapeutic peptides may be used in the treatment of SLE, traditionally used to treat SLE and this may offer a synergistic and drug-sparing effect.



# Highlight the Answer Passages

To help you read this paper, below is a list of key questions that are answered in this paper.

What condition does this paper study?

Systemic Lupus Erythematosus is a disease that affects about 5 million people in the world, mostly women.

see paragraph on page 2



How is the condition usually treated?

After you get the diagnosis of lupus, the doctor will see how bad your lupus is and how much it affects your body.

see paragraph on page 5

What did the paper want to find out?

This is a review of the evidence and reasons why doctors are using peptides to treat lupus.

see paragraph on page 3

What did the paper do?

The next paragraphs tell us about some drugs that are being tested to see if they can help people with lupus.

see paragraph on page 5

EXPERT OPINION ON INVESTIGATIONAL DRUGS  
2020, VOL. 29, NO. 8, 845–867  
<https://doi.org/10.1080/13543784.2020.1777983>



## REVIEW

## Therapeutic peptides for the treatment of systemic lupus erythematosus: a place in therapy

Rossella Talotta<sup>a</sup>, Fabiola Atzeni<sup>a</sup> and Magdalena Janina Laska<sup>b</sup>

<sup>a</sup>Department of Clinical and Experimental Medicine, Rheumatology Unit, Azienda Ospedaliera “Gaetano Martino”, University of Messina, Messina, Italy; <sup>b</sup>Department of Rheumatology, Aarhus University Hospital, Aarhus, Denmark

### ABSTRACT

**Introduction:** Studies *in vitro* and *in vivo* have identified several peptides that are potentially useful in treating systemic lupus erythematosus (SLE). The rationale for their use lies in the cost-effective production, high potency, target selectivity, low toxicity, and a peculiar mechanism of action that is mainly based on the induction of immune tolerance. Three therapeutic peptides have entered clinical development, but they have yielded disappointing results. However, some subsets of patients, such as those with the positivity of anti-dsDNA antibodies, appear more likely to respond to these medications.

**Areas covered:** This review evaluates the potential use of therapeutic peptides for SLE and gives an opinion on how they may offer advantages for SLE treatment.

**Expert opinion:** Given their acceptable safety profile, therapeutic peptides could be added to agents traditionally used to treat SLE and this may offer a synergistic and drug-sparing effect, especially in selected patient populations. Moreover, they could temporarily be utilized to manage SLE flares, or be administered as a vaccine in subjects at risk. Efforts to ameliorate bioavailability, increase the half-life and prevent immunogenicity are ongoing. The formulation of hybrid compounds, like peptibodies or peptidomimetic small molecules, is expected to yield renewed treatments with a better pharmacologic profile and increased efficacy.

### ARTICLE HISTORY

Received 23 January 2020

Accepted 1 June 2020

### KEYWORDS

Pathogenesis; systemic lupus erythematosus; therapeutic peptides; treatment; peptibodies; peptides

### 1. Introduction

Systemic lupus erythematosus (SLE) is the prototypical autoimmune connective tissue disease, affecting 5 million individuals worldwide, mainly women during the fertile age [1]. Clinical presentation broadly varies from patient to patient, with kidney and central nervous system (CNS) involvement representing the most severe complications [2]. The disease

and usually shows less successful results, *Figure 2* [8]. Several immunologic pathways are, in fact, concomitantly activated in SLE, and this justifies the use of medications like steroids, immunosuppressants and disease-modifying anti-rheumatic drugs (DMARDs), which unselectively counteract the immune response. Such a combo-therapy can indeed have many summing side effects that can be further

# Provide a Simplified Language Gist

see if they can help people with lupus.

see paragraph on [page 5](#)

What were the new treatment(s), if any, this paper looked into?

A peptide is a small molecule made up of amino acids, which are the building blocks of proteins.

see paragraph on [page 3](#)

What did the paper find?

In the future, scientists will make new drugs that will help people with lupus.

see paragraph on [page 18](#), [page 20](#), [page 19](#)

Are the findings different depending on a person's demographics?

These new drugs are designed to target the bad proteins that cause SLE.

see paragraph on [page 4](#), [page 17](#)

What are the limitations of the findings?

Even though the medicine worked well in the lab, it did not work as well in real life

see paragraph on [page 18](#)

Cycle cells may unleash, through an impaired phagocytosis, the cascade of events leading to type I IFN production, and amplify, through this pathway, inflammation and tissue damage.

## Conclusion

Better knowledge of the pathogenesis of SLE is expected to enrich the therapeutic armamentarium and facilitate the management of the disease. The use of peptides, specifically designed to target SLE-related epitopes or crucial pathways, may represent a novel fascinating opportunity. Given their good safety profile and immunomodulatory properties, therapeutic peptides could be added to standard of care, and, perhaps, allow the sparing of conventional drugs. In addition, their prescription might be tailored to specific subsets of patients having the highest likelihood of response. Nevertheless, despite the successful results observed in preclinical studies, RCTs showed a controversial efficacy profile concerning the use of these compounds in SLE. It is expected that future research, aiming at the amelioration of their physicochemical properties and at the improvement in the design of clinical trials, will bring more encouraging data on this innovative therapeutic panorama.

## Expert opinion

The treatment of SLE still relies on a conservative approach, combining multiple unselective immunosuppressive agents [8] and, consecutively, increasing the risk of unwanted side effects. Unlike other rheumatic diseases, the licensed use of biologic agents, which electively inhibit a specific target, has been solely limited to belimumab. Rituximab failed to achieve the primary endpoints in RCTs conducted in SLE patients [12,13,143] but, due to encouraging real-life data [11], its off-label use is advised in resistant severe manifestations [8]. The potential use of novel biological agents and small molecules is under investigation. Furthermore, several preliminary data on other small

peptide sequence by means of recombinant DNA technology and then modified in order to enhance their pharmacologic properties. Most of therapeutic peptides designed for SLE treatment are synthesized on the basis of immunodominant epitope sequences that are pathogenic in SLE alone. Therefore, they exert an immunomodulatory effect on autoreactive pDC and lymphocytes, without affecting, instead, the immune response against pathogens [26].

To date, no therapeutic peptide has been licensed and marketed for the use in SLE patients. The 21-mer peptide P140 is the only one entering phase III RCTs, and, despite controversial results [36,40,41,43], its development is still ongoing. Edratide, synthesized on the basis of the hCDR1 expressing the major idiotype 16/6 Id, showed promising results in preclinical studies [48,50,54,55], but failed to meet the primary endpoint in a phase II RCT [57], with the following interruption of further clinical development. Both the two peptides appeared more effective in anti-dsDNA seropositive patients, and, although data are lacking, a beneficial role may be supposed in those SLE manifestations related to anti-dsDNA antibodies, such as glomerulonephritis.

**The gist:** A drug that targets a specific part of the immune system is being tested to see if it can help people with a disease called lupus. The drug is not yet approved, but it has been tested in people with lupus and seems to help them.

*Generated automatically*

may be translated in humans is an uncertain issue that needs to be addressed in future research.

The discovery of novel molecular targets is expected to enrich, in the next years, the panorama of therapeutic peptides for SLE, among which CXCR4 and STING antagonists and virus-derived peptides seem promising candidates [116,119,135].

Meanwhile, research is focusing on the optimization of the physicochemical structure of preexisting peptides with the aim of enhancing their pharmacologic properties, including bioavailability and half-life. One of these efforts consists of the

Resume where you left by pressing the browser button

# Provide a Simplified Section Summary

see if they can help people with lupus.

see paragraph on [page 5](#)

What were the new treatment(s), if any, this paper looked into?

A peptide is a small molecule made up of amino acids, which are the building blocks of proteins.

see paragraph on [page 3](#)

What did the paper find?

In the future, scientists will make new drugs that will help people with lupus.

see paragraph on [page 18](#), [page 20](#), [page 19](#)

Are the findings different depending on a person's demographics?

These new drugs are designed to target the bad proteins that cause SLE.

see paragraph on [page 4](#), [page 17](#)

What are the limitations of the findings?

Even though the medicine worked well in the lab, it did not work as well in real life

see paragraph on [page 18](#)

release of IL-6 and IFN $\gamma$  following the stimulation with mitogens in PBMCs of patients with SLE. Additionally, when subcutaneously given at a dose of 10–100 mg/kg 5 days a week to MRL/lpr mice, it increased life-span and reduced proteinuria, though the titers of anti-DNA antibodies and the histological grading of glomerulonephritis remained unchanged. Although promising, research in this field was discontinued.

## 2.3. Peptibodies

**Section summary:** A medicine called AMG623 is a drug that is made to treat a disease called lupus. The medicine is made of four pieces of a protein that binds to BLYS, which is a protein that makes lupus worse. The medicine is made of pieces of a protein that binds to BLYS, and it's attached to the Fc part of an antibody.

*Generated automatically*

## 3.1. Blisibimod

AMG623 is a peptibody carrying four high-affinity BLYS-binding peptides grafted onto the Fc portion of a IgG $_1$  [97]. The compound was electively designed for SLE on the basis of the results of a phage display library screening, yielding a group of peptides binding to both soluble and membrane-bound BLYS with high affinity [98]. BLYS plays a crucial role in SLE pathogenesis, as it is able to foment B cell auto-reactivity, being also strictly interconnected with the type I IFN pathway. In preclinical experiments [99], the compound constrained the number of peripheral and spleen B lymphocytes in healthy BALB/c mice, and these results were confirmed in a BLYS-mediated B lymphocyte proliferation assay. Using NZBxNZW

appeared to decrease peripheral B lymphocyte count and unbalance the memory/naive B cell ratio at the detriment of the latter, though without any significant association with immunogenicity.

The phase II PEARL-SC study (ClinicalTrials.gov ID NCT01162681) aimed to assess the efficacy and safety of s.c. blisibimod at three different dose regimens (100 mg every week, 200 mg every week or 200 mg every 4 weeks) in 547 recruited SLE patients [101]. An active disease at baseline corresponding to a safety of estrogens in lupus erythematosus

ment (SELENA)-SLEDAI score  $\geq 6$  and the re renal and CNS complications represented for the study. The primary endpoint, consistement of the SRI-5 response at week 24, was ver, the dose of 200 mg administered once ntly reduced fatigue [102], and tended to spare the use of glucocorticoids and amelioral and immunologic domains, especially in vere disease activity. The safety profile was ADA not detected. Following the completion of this study, 32 participants entered the open-label extension phase (PEARL-OLE; ClinicalTrials.gov ID NCT01305746) [103]. This trial further confirmed the beneficial effect of blisibimod on proteinuria, anti-dsDNA antibody titers, and hypocomplementemia and reassured on the long-term safety profile.

The efficacy and safety profile of blisibimod was then tested in three phase III RCTs. The RCT CHABLIS-SC1 (ClinicalTrials.gov ID NCT01395745) was designed in order to selectively target those SLE subsets of patients who appeared more likely to respond to the drug in the previous phase II study. A total of 442 SLE patients were enrolled and randomized to receive blisibimod 200 mg subcutaneously injected once weekly or placebo plus standard of care. The primary

# Usability study

**Participants:** 24 non-expert readers.

**Reading task:** Read a paper for 10 minutes. Then, answer a set of questions about the paper.

**Conditions (×2):** Each reader completed tasks with 2 of 4 interface variants (one variant was a featureless baseline).

# Usability Study Results

**Baseline PDF reader:** Most participants read papers linearly and spent substantial time in dense sections with limited important information.

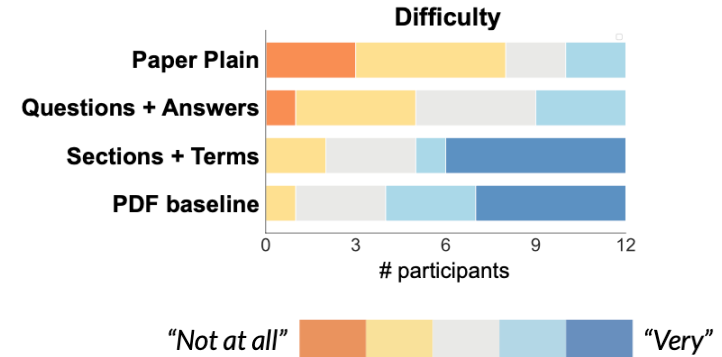
**Paper Plain:** All participants reached the end of the paper.  
Section gists and term definitions helped read dense passages.  
The questions & answer gists supported quick navigation and helped decide which sections to read



# The effect on understanding papers

Participants reported it was **less difficult** to read the paper and that they **better understood** the paper, when using Paper Plain.

That said, participants answered questions about the paper **about as correctly** with both Paper Plain and the baseline. They tended to answer more correctly when questions were answerable by clicking questions in the key question index.



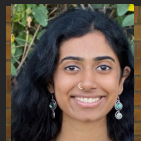
# INTELLIGENT PAPER SCANNING

SCIM: Intelligent Skimming Support

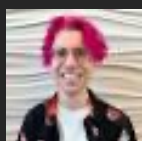
@ACM IUI '23



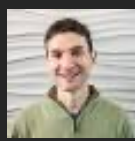
Raymond Fok



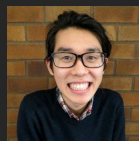
Hita  
Kambhamettu



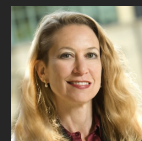
Luca  
Soldaini



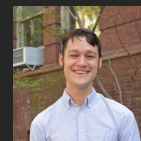
Jonathan  
Bragg



Kyle  
Lo



Marti A. Hearst



Andrew Head



Daniel S. Weld

# Lots of content – where to start reading?

## Linguistically-Informed Self-Attention for Semantic Role Labeling

Emma Strubell<sup>1</sup>, Patrick Verga<sup>1</sup>, Daniel Andor<sup>2</sup>, David Weiss<sup>2</sup> and Andrew McCallum<sup>1</sup>

<sup>1</sup>College of Information and Computer Sciences  
University of Massachusetts Amherst  
{strubell, pat, mcallum}@cs.umass.edu  
<sup>2</sup>Google AI Language  
New York, NY  
{andor, djweiss}@google.com

### Abstract

Current state-of-the-art semantic role labeling (SRL) uses a deep neural network with no explicit linguistic features. However, prior work has shown that gold syntax trees can dramatically improve SRL decoding, suggesting the possibility of increased accuracy from explicit modeling of syntax. In this work, we present linguistically-informed self-attention (LISA): a neural network model that combines multi-head self-attention with multi-task learning across dependency parsing, part-of-speech tagging, predicate detection and SRL. Unlike previous models which require significant pre-processing to prepare linguistic features, LISA can incorporate syntax using merely raw tokens as input, encoding the sequence only once to simultaneously perform parsing, predicate detection and role labeling for all predicates. Syntax is incorporated by training one attention head to attend to syntactic parents for each token. Moreover, if a high-quality syntactic parse is already available, it can be beneficially injected at test time without re-training our SRL model. In experiments on CoNLL-2005 SRL, LISA achieves new state-of-the-art performance for a model using predicted predicates and standard word embeddings, attaining 2.5 F1 absolute higher than the previous state-of-the-art on newswire and more than 3.5 F1 on out-of-domain data, nearly 10% reduction in error. On CoNLL-2012 English SRL we also show an improvement of more than 2.5 F1. LISA also out-performs the state-of-the-art with contextually-encoded (ELMo) word representations, by nearly 1.0 F1 on news and more than 2.0 F1 on out-of-domain text.

### 1 Introduction

Semantic role labeling (SRL) extracts a high-level representation of meaning from a sentence, labeling e.g. who did what to whom. Explicit representations of such semantic information have been

shown to improve results in challenging downstream tasks such as dialog systems (Tur et al., 2005; Chen et al., 2013), machine reading (Berant et al., 2014; Wang et al., 2015) and translation (Liu and Gildea, 2010; Bazzafshan and Gildea, 2013). Though syntax was long considered an obvious prerequisite for SRL systems (Levin, 1993; Pinyanov et al., 2008), recently deep neural network architectures have surpassed syntactically-informed models (Zhou and Xu, 2015; Marcheggiani et al., 2017; He et al., 2017; Tan et al., 2018; He et al., 2018), achieving state-of-the-art SRL performance with no explicit modeling of syntax. An additional benefit of these end-to-end models is that they require just raw tokens and (usually) detected predicates as input, whereas richer linguistic features typically require extraction by an auxiliary pipeline of models.

Still, recent work (Roth and Lapata, 2016; He et al., 2017; Marcheggiani and Titov, 2017) indicates that neural network models could see even higher accuracy gains by leveraging syntactic information rather than ignoring it. He et al. (2017) indicate that many of the errors made by a syntax-free neural network on SRL are tied to certain syntactic confusions such as prepositional phrase attachment, and show that while constrained inference using a relatively low-accuracy predicted parse can provide small improvements in SRL accuracy, providing a gold-quality parse leads to substantial gains. Marcheggiani and Titov (2017) incorporate syntax from a high-quality parser (Kiparwasser and Goldberg, 2016) using graph convolutional neural networks (Kipf and Welling, 2017), but like He et al. (2017) they attain only small increases over a model with no syntactic parse, and even perform worse than a syntax-free model on out-of-domain data. These works suggest that though syntax has the potential to improve neural network SRL models, we have not

and normalize with the softmax function to produce a distinct distribution for each token over all the tokens in the sentence:

$$A_i^{(j)} = \text{softmax}(d_i^{(j)} \odot q_i^{(j)} K_i^{(j)T}) \quad (2)$$

These attention weights are then multiplied by  $V_i^{(j)}$  for each token to obtain the self-attended-token representations  $M_i^{(j)}$ :

$$M_i^{(j)} = A_i^{(j)} V_i^{(j)} \quad (3)$$

Row  $i$  of  $M_i^{(j)}$ , the self-attended representation for token  $i$  at layer  $j$ , is thus the weighted sum with respect to  $i$  (with weights given by  $A_i^{(j)}$ ) over the token representations in  $V_i^{(j)}$ .

The outputs of all attention heads for each token are concatenated, and this representation is passed to the feed-forward layer, which consists of two linear projections each followed by leaky ReLU activations (Maas et al., 2013). We add the output of the feed-forward to the initial representation and apply layer normalization to give the final output of self-attention layer  $j$ , as in Eqn. 1.

### 2.2 Syntactically-informed self-attention

Typically, neural attention mechanisms are left on their own to learn to attend to relevant inputs. Instead, we propose training the self-attention to attend to specific tokens corresponding to the syntactic structure of the sentence as a mechanism for passing linguistic knowledge to later layers.

Specifically, we replace one attention head with the deep bi-affine model of Draz and Manning (2017), trained to predict syntactic dependencies. Let  $A_{\text{parse}}$  be the parse attention weights, at layer  $l$ . Its input is the matrix of token representations  $S^{(l-1)}$ . As with the other attention heads, we project  $S^{(l-1)}$  into key, value and query representations, denoted  $K_{\text{parse}}, Q_{\text{parse}}, V_{\text{parse}}$ . Here the key and query projections correspond to *parent* and *dependent* representations of the tokens, and we allow their dimensions to differ from the rest of the attention heads to more closely follow the implementation of Draz and Manning (2017). Unlike the other attention heads which use a dot product to score key-query pairs, we score the compatibility between  $K_{\text{parse}}$  and  $Q_{\text{parse}}$  using a bi-affine operator  $U_{\text{heads}}$  to obtain attention weights:

$$A_{\text{parse}} = \text{softmax}(Q_{\text{parse}} U_{\text{heads}} K_{\text{parse}}^T) \quad (4)$$

These attention weights are used to compose a weighted average of the value representations  $V_{\text{parse}}$  as in the other attention heads. We apply auxiliary supervision at this attention head to encourage it to attend to each token's parent in a syntactic dependency tree, and to encode information about the token's dependency label. Denoting the attention weight from token  $i$  to a candidate head  $q$  as  $A_{\text{parse}}[i, q]$ , we model the probability of token  $i$  having parent  $q$  as:

$$P(q = \text{head}(i) \mid X) = A_{\text{parse}}[i, q] \quad (5)$$

using the attention weights  $A_{\text{parse}}[i, q]$  as the distribution over possible heads for token  $i$ . We define the root token as having a self-loop. This attention head thus emits a directed graph<sup>1</sup> where each token's parent is the token to which the attention  $A_{\text{parse}}$  assigns the highest weight.

We also predict dependency labels using per-class bi-affine operations between parent and dependent representations  $Q_{\text{parse}}$  and  $K_{\text{parse}}$  to produce per-label scores, with locally normalized probabilities over dependency labels  $y_i^{(j)}$  given by the softmax function. We refer the reader to Draz and Manning (2017) for more details.

This attention head now becomes an oracle for syntax, denoting  $P$ , providing a dependency parse to downstream layers. This model not only predicts its own dependency axes, but allows for the injection of auxiliary parse information at test time by simply setting  $A_{\text{parse}}$  to the parse parents produced by e.g. a state-of-the-art parser. In this way, our model can benefit from improved, external parsing models without re-training. Unlike typical multi-task models, ours maintains the ability to leverage external syntactic information.

### 2.3 Multi-task learning

We also share the parameters of lower layers in our model to predict POS tags and predicates. Following He et al. (2017), we focus on the end-to-end setting, where predicates must be predicted on-the-fly. Since we also train our model to predict syntactic dependencies, it is beneficial to give the model knowledge of POS information. While much previous work employs a pipelined approach to both POS tagging for dependency parsing and predicate detection for SRL, we take a multi-task learning (MTL) approach (Caruana,

<sup>1</sup>Usually the head emits a tree, but we do not enforce it here.

GloVe	Dev			WSI Test			Brown Test		
	P	R	F1	P	R	F1	P	R	F1
He et al. (2017) PoE	81.8	81.2	81.5	82.0	83.4	82.7	69.7	70.3	70.1
He et al. (2018)	81.3	81.9	81.6	81.2	83.9	82.5	69.7	71.9	70.8
SA	83.52	81.28	82.39	84.17	83.28	83.72	72.98	70.1	71.51
LISA	83.1	81.39	82.24	84.07	83.16	83.61	73.32	70.56	71.91
+D&M	<b>84.59</b>	<b>82.59</b>	<b>83.58</b>	<b>85.53</b>	<b>84.45</b>	<b>84.99</b>	<b>75.8</b>	<b>73.54</b>	<b>74.66</b>
+Gold	87.91	85.73	86.81	—	—	—	—	—	—
ELMo									
He et al. (2018)	84.9	<b>85.7</b>	85.3	84.8	<b>87.2</b>	86.0	73.9	<b>78.4</b>	76.1
SA	85.78	84.74	85.26	86.21	85.98	86.09	77.1	75.61	76.35
LISA	<b>86.07</b>	84.64	<b>85.35</b>	86.69	86.42	86.55	78.95	77.17	78.05
+D&M	85.83	84.51	85.17	<b>87.13</b>	<b>86.67</b>	<b>86.90</b>	<b>79.02</b>	<b>77.49</b>	<b>78.25</b>
+Gold	88.51	86.77	87.63	—	—	—	—	—	—

Table 1: Precision, recall and F1 on the CoNLL-2005 development and test sets.

WSI Test	P	R	F1
He et al. (2018)	84.2	83.7	83.9
Tan et al. (2018)	84.5	85.2	84.8
SA	84.7	84.24	84.47
LISA	84.72	84.57	84.64
+D&M	<b>86.02</b>	<b>86.05</b>	<b>86.04</b>
Brown Test	P	R	F1
He et al. (2018)	74.2	73.1	73.7
Tan et al. (2018)	73.5	74.6	74.1
LISA	73.89	72.39	73.13
LISA	74.77	74.32	74.55
+D&M	<b>76.65</b>	<b>76.44</b>	<b>76.54</b>

Table 2: Precision, recall and F1 on CoNLL-2005 with gold predicates.

### 4.1 Semantic role labeling

Table 1 lists precision, recall and F1 on the CoNLL-2005 development and test sets using predicted predicates. For models using GloVe embeddings, our syntax-free SA model already achieves a new state-of-the-art by jointly predicting predicates, POS and SRL. LISA with its own parses performs comparably to SA, but when supplied with D&M parses LISA out-performs the previous state-of-the-art by 2.5 F1 points. On the out-of-domain Brown test set, LISA also performs comparably to its syntax-free counterpart with its own parses, but with D&M parses LISA performs exceptionally well, more than 3.5 F1 points higher than He et al. (2018). Incorporating ELMo em-

beddings improves all scores. The gap in SRL F1 between models using LISA and D&M parses is smaller due to LISA's improved parsing accuracy (see §4.2), but LISA with D&M parses still achieves the highest F1: nearly 1.0 absolute F1 higher than the previous state-of-the-art on WSI, and more than 2.0 F1 higher on Brown. In both settings LISA leverages domain-agnostic syntactic information rather than over-fitting to the newswire training data, which leads to high performance even on out-of-domain text.

To compare to more prior work we also evaluate our models in the artificial setting where gold predicates are provided at test time. For fair comparison we use GloVe embeddings, provide predicate identifier embeddings on the input and re-encode the sequence relative to each gold predicate. Here LISA still excels: with D&M parses, LISA out-performs the previous state-of-the-art by more than 2 F1 on both WSI and Brown.

Table 3 reports precision, recall and F1 on the CoNLL-2012 test set. We observe performance similar to that observed on CoNLL-2005: Using GloVe embeddings our SA baseline already out-performs He et al. (2018) by nearly 1.5 F1. With its own parses, LISA slightly under-performs our syntax-free model, but when provided with stronger D&M parses LISA out-performs the state-of-the-art by more than 2.5 F1. Like CoNLL-2005, ELMo representations improve all models and close the F1 gap between models supplied with LISA and D&M parses. On this dataset ELMo also substantially narrows the



# Idea: Highlight Key Passages by Rhetorical Purpose

## Recommendation Unlearning

Chong Chen<sup>1\*</sup>, Fei Sun<sup>2†</sup>, Min Zhang<sup>1†</sup>, Bolin Ding<sup>2</sup>

<sup>1</sup>Department of Computer Science and Technology, Institute for Artificial Intelligence,  
Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

<sup>2</sup>DAMO Academy, Alibaba Group

cc17@mails.tsinghua.edu.cn, z-m@tsinghua.edu.cn, <sup>2</sup>{ofey.sf, bolin.ding}@alibaba-inc.com

### ABSTRACT

Recommender systems provide essential web services by learning users' personal preferences from collected data. However, in many cases, systems also need to forget some training data. From the perspective of privacy, users desire a tool to erase the impacts of their sensitive data from the trained models. From the perspective of utility, if a system's utility is damaged by some bad data, the system needs to forget such data to regain utility. While unlearning is very important, it has not been well-considered in existing recommender systems. Although there are some researches have studied the problem of machine unlearning, existing methods can not be directly applied to recommendation as they are unable to consider the collaborative information.

In this paper, we propose RecEraser, a general and efficient machine unlearning framework tailored to recommendation tasks. The main idea of RecEraser is to divide the training set into multiple shards and train submodels with these shards. Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. We then further propose an adaptive aggregation method to improve the global model utility. Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. The source code can be found at <https://github.com/chenchongthu/Recommendation-Unlearning>.

### CCS CONCEPTS

• Information systems → Recommender systems; • Security and privacy → Privacy protections.

### KEYWORDS

Machine Unlearning; Selective Deletion; Recommender Systems; Collaborative Filtering;

### ACM Reference Format:

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation Unlearning. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*, April 25–29, 2022, Virtual Event, Lyon, France. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3485447.3511997>

### 1 INTRODUCTION

Recommender systems provide personalized service for users to alleviate the information overload problem, playing a more and more important role in a wide range of applications, such as e-commerce [14, 38], social media [8, 10, 16, 48], and news portal [41]. The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings).

Once a recommender system is built, it has potentially memorized the training data. However, in many cases, a recommender system also needs to forget certain sensitive data and its complete lineage, which is called *Recommendation Unlearning* in this paper. Consider privacy first, recent researches have shown that users' sensitive information could be leaked from the trained models, e.g., recommender systems [30], big pretrained [4] and finetuned natural language models [49]. In such cases, users desire a tool to erase the impacts of their sensitive information from the trained models. The second reason is utility. Nowadays, new data is collected incrementally to further refine existing models [31]. However, bad data (or called dirty data), e.g., polluted data in poisoning attacks [33] or out-of-distribution (OOD) data [3], will seriously degrade the performance of recommendation. Once these data are identified, the system needs to forget them to regain utility. Moreover, generally users' preferences are dynamic and changeable [44]. For example, a user who wants to buy a mobile phone will be happy to see recommendations about mobile phones. But after the purchase, she/he will not be interested in the recommendations of new mobile phones for a period of time. In this case, the user will want to delete certain data so that the system can provide more useful recommendations.

The most naive unlearning method is to retrain from the original data after removing the samples that need to be forgotten. Unfortunately, the difficulty in applying this method lies in the expensive computational cost for large-scale data. Some efforts have been devoted to resolving the inefficiency issue of machine unlearning in the domains of computer vision and natural language process-

AI-generated highlighting to support skimming

Settings

**Method** Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. **Page 1**

**Result** We then further propose an adaptive aggregation method to improve the global model utility. **Page 1**

**Result** Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. **Page 1**

**Method** The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings). **Page 1**

**Method** However, in many cases, a recommender system also needs to forget certain sensitive data and its complete lineage, which is called

Feedback? [Contact Us](#)

arXiv:2201.06820v2 [cs.LG] 25 Jan 2022

Objective

Method

Result

# Recommendation Unlearning

Chong Chen<sup>1\*</sup>, Fei Sun<sup>2†</sup>, Min Zhang<sup>1†</sup>, Bolin Ding<sup>2</sup>

<sup>1</sup>Department of Computer Science and Technology, Institute for Artificial Intelligence,  
Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

<sup>2</sup>DAMO Academy, Alibaba Group

cc17@mails.tsinghua.edu.cn, z-m@tsinghua.edu.cn, <sup>2</sup>{ofey.sf, bolin.ding}@alibaba-inc.com

## ABSTRACT

Recommender systems provide essential web services by learning users' personal preferences from collected data. However, in many cases, systems also need to forget some training data. From the perspective of privacy, users desire a tool to erase the impacts of their sensitive data from the trained models. From the perspective of utility, if a system's utility is damaged by some bad data, the system needs to forget such data to regain utility. While unlearning is very important, it has not been well-considered in existing recommender systems. Although there are some researches have studied the problem of machine unlearning, existing methods can not be directly applied to recommendation as they are unable to consider the collaborative information.

In this paper, we propose RecEraser, a general and efficient machine unlearning framework tailored to recommendation tasks. The main idea of RecEraser is to divide the training set into multiple shards and train submodels with these shards. Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. We then further propose an adaptive aggregation method to improve the global model utility. Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. The source code can be found at <https://github.com/chenchongthu/Recommendation-Unlearning>

## ACM Reference Format:

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation Unlearning. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*, April 25–29, 2022, Virtual Event, Lyon, France. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3485447.3511997>

## 1 INTRODUCTION

Recommender systems provide personalized service for users to alleviate the information overload problem, playing a more and more important role in a wide range of applications, such as e-commerce [14, 38], social media [8, 10, 16, 48], and news portal [41]. The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings).

Once a recommender system is built, it has potentially memorized the training data. However, in many cases, a recommender system also needs to forget certain sensitive data and its complete lineage, which is called *Recommendation Unlearning* in this paper. Consider privacy first, recent researches have shown that users' sensitive information could be leaked from the trained models, e.g., recommender systems [50], big pretrained [4] and finetuned natural language models [49]. In such cases, users desire a tool to erase the impacts of their sensitive information from the trained models. The second reason is utility. Nowadays, new data is collected incrementally to further refine existing models [51]. However, bad data (or called dirty data), e.g., polluted data in poisoning attacks [33]

25 Jan 2022

Objective

Method

Result

Method

Method

Result

# Recommendation Unlearning

Chong Chen<sup>1\*</sup>, Fei Sun<sup>2†</sup>, Min Zhang<sup>1†</sup>, Bolin Ding<sup>2</sup>

<sup>1</sup>Computer Science and Technology, Institute for Artificial Intelligence,  
for Information Science and Technology, Tsinghua University, Beijing 100084, China

<sup>2</sup>DAMO Academy, Alibaba Group  
z-m@tsinghua.edu.cn, <sup>2</sup>{ofey.sf, bolin.ding}@alibaba-inc.com

## ACM Reference Format:

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation Unlearning. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*, April 25–29, 2022, Virtual Event, Lyon, France. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3485447.3511997>

## 1 INTRODUCTION

Recommender systems provide personalized service for users to alleviate the information overload problem, playing a more and more important role in a wide range of applications, such as e-commerce [14, 38], social media [8, 10, 16, 48], and news portal [41]. The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings).

Once a recommender system is built, it has potentially memorized the training data. However, in many cases, a recommender system also needs to forget certain sensitive data and its complete lineage, which is called *Recommendation Unlearning* in this paper. Consider privacy first, recent researches have shown that users' sensitive information could be leaked from the trained models, e.g., recommender systems [50], big pretrained [4] and finetuned natural language models [49]. In such cases, users desire a tool to erase the impacts of their sensitive information from the trained models. The second reason is utility. Nowadays, new data is collected incrementally to further refine existing models [51]. However, bad data (or called dirty data), e.g., polluted data in poisoning attacks [33]

Method

Method

Result

## Skimming Highlights



AI-generated highlighting to support skimming

Settings

**Method** Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. **Page 1**

**Result** We then further propose an adaptive aggregation method to improve the global model utility. **Page 1**

**Result** Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. **Page 1**

**Method** The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42].



# Recommendation Unlearning

Chong Chen<sup>1\*</sup>, Fei Sun<sup>2†</sup>, Min Zhang<sup>1†</sup>, Bolin Ding<sup>2</sup>

<sup>1</sup>Department of Computer Science and Technology, Institute for Artificial Intelligence,  
Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China

<sup>2</sup>DAMO Academy, Alibaba Group  
cc17@mails.tsinghua.edu.cn, z-m@tsinghua.edu.cn, {ofey.sf, bolin.ding}@alibaba-inc.com

## ABSTRACT

Recommender systems provide essential web services by learning users' personal preferences from collected data. However, in many cases, systems also need to forget some training data. From the perspective of privacy, users desire a tool to erase the impacts of their sensitive data from the trained models. From the perspective of utility, if a system's utility is damaged by some bad data, the system needs to forget such data to regain utility. While unlearning is very important, it has not been well-considered in existing recommender systems. Although there are some researches have studied the problem of machine unlearning, existing methods can not be directly applied to recommendation as they are unable to consider the collaborative information.

In this paper, we propose RecEraser, a general and efficient machine unlearning framework tailored to recommendation tasks. The main idea of RecEraser is to divide the training set into multiple shards and train submodels with these shards. Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. We then further propose an adaptive aggregation method to improve the global model utility. Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. The source code can be found at <https://github.com/chenchongthu/Recommendation-Unlearning>

## CCS CONCEPTS

• Information systems → Recommender systems; • Security and privacy → Privacy protections.

## KEYWORDS

Machine Unlearning; Selective Deletion; Recommender Systems; Collaborative Filtering.

## ACM Reference Format:

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation Unlearning. In *Proceedings of the ACM Web Conference 2022 (WWW '22)*, April 25–29, 2022, Virtual Event, Lyon, France. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3485447.3511997>

## 1 INTRODUCTION

Recommender systems provide personalized service for users to alleviate the information overload problem, playing a more and more important role in a wide range of applications, such as e-commerce [14, 38], social media [8, 10, 16, 48], and news portal [41]. The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings).

Once a recommender system is built, it has potentially memorized the training data. However, in many cases, a recommender system also needs to forget certain sensitive data and its complete lineage, which is called *Recommendation Unlearning* in this paper. Consider privacy first, recent researches have shown that users' sensitive information could be leaked from the trained models, e.g., recommender systems [50], big pretrained [4] and finetuned natural language models [49]. In such cases, users desire a tool to erase the impacts of their sensitive information from the trained models. The second reason is utility. Nowadays, new data is collected incrementally to further refine existing models [51]. However, bad data (or called dirty data), e.g., polluted data in poisoning attacks [33] or out-of-distribution (OOD) data [3], will seriously degrade the performance of recommendation. Once these data are identified, the system needs to forget them to regain utility. Moreover, generally users' preferences are dynamic and changeable [44]. For example, a user who wants to buy a mobile phone will be happy to see recommendations about mobile phones. But after the purchase, she/he will not be interested in the recommendations of new mobile phones for a period of time. In this case, the user will want to forget the information about mobile phones.

Method

Method

Result

## Skimming Highlights

AI-generated highlighting to support skimming

Settings

**Method** Specifically, to keep the collaborative information of the data, we first design three novel data partition algorithms to divide training data into balanced groups. **Page 1**

**Result** We then further propose an adaptive aggregation method to improve the global model utility. **Page 1**

**Result** Experimental results on three public benchmarks show that RecEraser can not only achieve efficient unlearning but also outperform the state-of-the-art unlearning methods in terms of model utility. **Page 1**

**Method** The key of personalized recommender systems is known as collaborative filtering [28, 31, 37, 42], which learns users' preference based on their historical records (e.g., views, clicks, and ratings). **Page 1**

**Method** However, in many cases, a recommender system also needs to

arXiv:2201.06820v2 [cs.LG] 25 Jan 2022

Objective

Method

Result

# Usability Studies

**In-lab study:** 19 participants, expertise in NLP

Within-participants; each used a plain PDF reader and Scim

1 hour duration

**Longitudinal Diary Study:** 12 participants, with expertise

They choose which version of the reader to use

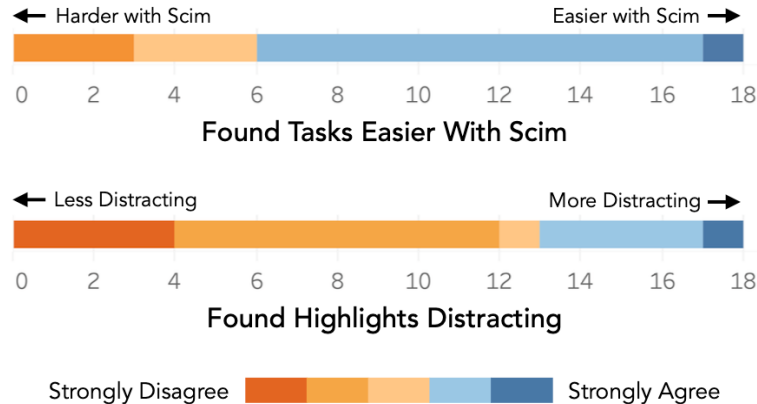
2 weeks

# In-lab Usability Study Results

Questions answered more quickly with Scim; No difference in accuracy

Most did not find highlights distracting

More akin to “scanning” than “skimming”



# Longitudinal Usability Study Results

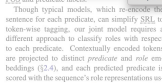
70% reported finding highlights useful

Assisted in focusing on important information

Provided a summary of the paper

Distracting when highlights are incorrect

# ScholarPhi

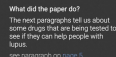


Specifically, consider the matrix  $S^{(j-1)}$  of  $\mathbb{T}$ -token representations at layer  $j-1$ . For each attention head  $h$ , we project this matrix into distinct

# Scim



# PaperPlain





# Doing this all on PDFs ...

Why? HTML would be easier, but PDFs are where the readers are (for now)

How? The rest of this talk.

# Talk Outline

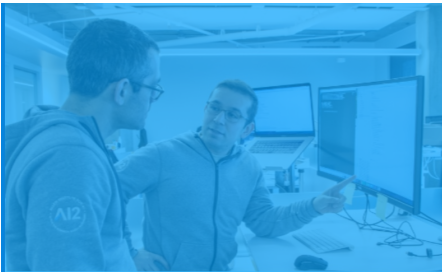


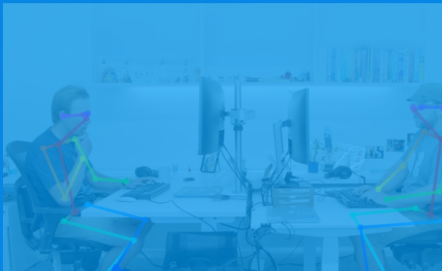

1. Three Motivating Demos

**2. AI2's Semantic Reader Project**

3. NLP and Document Analysis Techniques  
Including GPT-4 Experiments

# Allen Institute for AI (AI2)

A non-profit research institute  
dedicated to AI for the Common Good

 <p>Natural Language Processing</p>	 <p>Computer Vision</p>	<h3>Featured News</h3> <p>The AI2 ImpACT License Project: Open, Responsible AI Licenses for the Common Good</p> <p><a href="#">Learn more</a></p>
 <p>Semantic Scholar</p>	 <p>Research Visualization</p>	 <p>AI for the Environment</p>

# AI 2 Semantic Scholar Project



About Semantic Scholar

[Overview](#) [Our Team](#) [Publishers](#) [Blog](#) [AI2 Careers](#)

About Semantic Scholar

## Helping Scholars Discover New Insights

Semantic Scholar provides free, AI-driven search and discovery tools, and open resources for the global research community.

We index over 200 million academic papers sourced from [publisher partnerships](#), data providers, and web crawls.





50 person team  
7 year project

207M+ scientific paper index  
8M+ monthly active users

# Semantic Scholar Search



SEMANTIC SCHOLAR

document layout analysis

Search 🔍

Account ▾

About 498,000 results for “document layout analysis”

Fields of Study ▾

Date Range ▾

Has PDF

Author ▾

Journals & Conferences ▾

Sort by Relevance ▾



## PubLayNet: Largest Dataset Ever for Document Layout Analysis

[Xu Zhong](#), [Jianbin Tang](#), [Antonio Jimeno-Yepes](#) · Computer Science · IEEE International Conference on Document... · 16 August 2019

**TLDR** The PubLayNet dataset for **document layout analysis** is developed by automatically matching the XML representations and the content of over 1 million PDF articles that are publicly available on PubMed Central and demonstrated that deep neural networks trained on Pub LayNet accurately recognize the **layout** of scientific articles. [Expand](#)

👍 258 📄 PDF · 🖨 View on IEEE 📌 Save 🔔 Alert 🗣 Cite

## Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis

[Siwen Luo](#), [Yi Ding](#), [Siqu Long](#), [S. Han](#), [Josiah Poon](#) · Computer Science · International Conference on Computational... · 22 August 2022

**TLDR** The Doc-GCN presents an effective way to harmonize and integrate heterogeneous aspects for **Document Layout Analysis**, and achieves state-of-the-art results on three widely used DLA datasets: PubLayNet, FUNSD, and DocBank. [Expand](#)

👍 4 📄 PDF · 🖨 View on ACL 📌 Save 🔔 Alert 🗣 Cite



# Semantic Scholar API

Semantic Scholar API - Overview

## Providing a reliable source of scholarly data for developers

Build projects that accelerate scientific progress with the Semantic Scholar Academic Graph API

- [Code Examples](#)
- [Q&A Bank](#)
- [Request an API Key](#)
- [API Service Status Page](#)
- [Join th](#)
- [Read t](#)



Access Our Continually Updating Corpus

**206 Million**

Papers

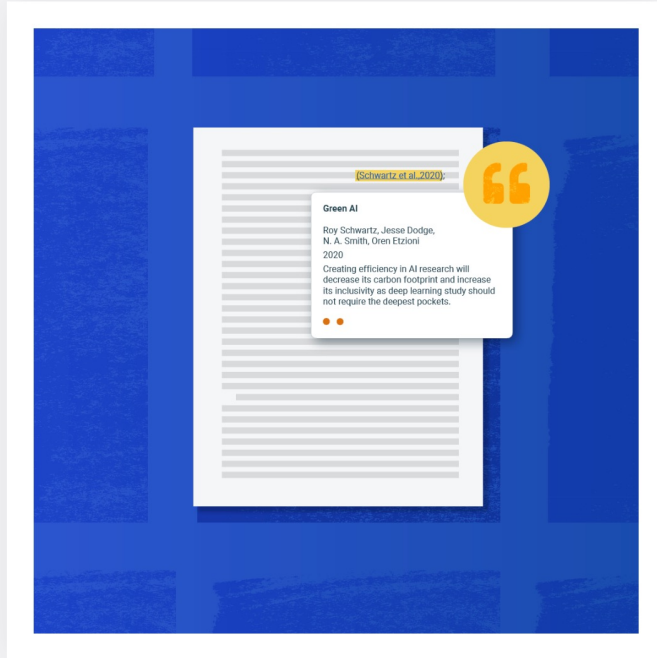
**2.49 Billion**

Citations

**79 Million**

Authors

# Semantic Reader Project



SEMANTIC READER

## Introducing Semantic Reader

An AI-Powered Augmented Scientific Reading  
Application

# Semantic Reader In Action

These solutions address the memory limitation problem, but not the communication overhead. In this paper, we address all of the aforementioned problems, by designing A Lite BERT (ALBERT) architecture that has significantly fewer parameters than a traditional BERT architecture.

ALBERT incorporates two parameter reduction techniques that lift the major obstacles in scaling pre-trained models. The first one is a factorized embedding parameterization. By decomposing the large vocabulary embedding matrix into two small matrices, we separate the size of the hidden layers from the size of vocabulary embedding. This separation makes it easier to grow the hidden size without significantly increasing the parameter size of the vocabulary embeddings. The second technique is cross-layer parameter sharing. This technique prevents the parameter from growing with the depth of the network. Both techniques significantly reduce the number of parameters for BERT without seriously hurting performance, thus improving parameter-efficiency. An ALBERT configuration similar to BERT-large has 18x fewer parameters and can be trained about 1.7x faster. The parameter reduction techniques also act as a form of regularization that stabilizes the training and helps with generalization.

To further improve the performance of ALBERT, we also introduce a self-supervised loss for sentence-order prediction (SOP). SOP primarily focuses on inter-sentence coherence and is designed to address the ineffectiveness (Yang et al., 2019; Liu et al., 2019) of the next sentence prediction (NSP) loss proposed in the original BERT.

As a result of these design decisions, we establish new state-of-the-art results for natural language understanding. SQuAD v1.1 benchmark to 89.4, and the F1 score on

## 2 RELATED WORK

### 2.1 SCALING UP REPRESENTATION

Learning representations of natural language tasks and has been widely adopted (Mikolov et al., 2013; Devlin et al., 2018).

#### XLNet: Generalized Autoregressive Pretraining for Language Understanding

Zhilin Yang, Zihang Dai, +3 authors · Quoc V. Le · Neural Information Processing Systems · 19 June 2019

TLDR XLNet is proposed, a generalized autoregressive pretraining method that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order and overcomes the limitations of BERT thanks to its autoregressive formulation.

# Currently only works on latex-backed PDFs

performance. For example, Devlin et al. (2018) shows that for natural language understanding tasks, using larger hidden layers leads to better performance. However, scaling up the model size and computation cost problems.

It is difficult to experiment with large models due to computational constraints, especially in terms

Save To Library Create Alert Cite

# Semantic Reader Project

## Powered by State-of-the-Art Research

Semantic Reader is based on research from the Semantic Scholar team at AI2, UC Berkeley and the University of Washington, and supported in part by the Alfred P. Sloan Foundation.

### Augmenting Scientific Papers with Just-in-Time, Position-Sensitive Definitions of Terms and Symbols

September 2020 CHI

**TLDR** This work introduces ScholarPhi, an augmented reading interface with four novel features: tooltips that surface position-sensitive definitions from elsewhere in a paper, a filter it to reveal h across the p diagrams the parallel, and glossary of ii

### CiteRead: Integrating Localized Citation Contexts into Scientific Paper Reading

March 2022 IUI

**TLDR** A novel paper reading experience that integrates relevant information about follow-on work directly into a paper, allowing readers to learn about newer papers and see

### Scim: Intelligent Faceted Highlights for Interactive, Multi-Pass Skimming of Scientific Papers

May 2022 ArXiv

**TLDR** Scim is presented, an AI-augmented reading interface designed to help researchers skim papers by automatically identifying, classifying, and highlighting rhetorical tion needs.

**But ... it is a challenge to translate some of this research into the product**

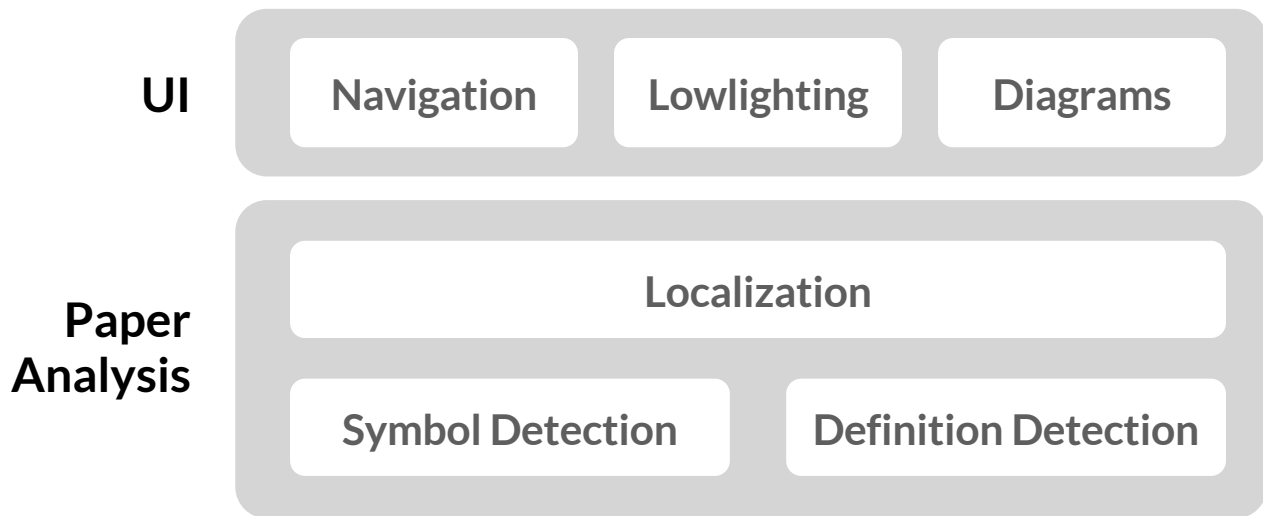
# Talk Outline

1. Three Motivating Examples of UIs

2. AI2's Semantic Reader Project

**3. NLP and Document Analysis Techniques**  
Including GPT-4 Experiments

# The construction of ScholarPhi



**Major challenge: fine-grained locations of symbols**



# The construction of ScholarPhi

**Major challenge: fine-grained locations of symbols**

**Solution: Link latex code to location in a PDF image**

# Identifying bounding boxes for symbols

LaT<sub>E</sub>X

function:\$P(y\_t^{\text{prp}}\$  
...\$, where...

function:  
`{\color{orange}}`\$P(y\_t^{\text{prp}}\$  
...\$), where...

Images of PDF

function:  $P(y_t^{\text{prp}} | \mathcal{X}) \propto \exp(r_t)$ , where

function:  $P(y_t^{\text{prp}} | \mathcal{X}) \propto \exp(r_t)$ , where

---

$$P(y_t^{\text{prp}} | \mathcal{X}) \propto \exp(r_t)$$

# Identifying bounding boxes for symbols

Equation

$$E_{f_d} = x^n$$

Contains 8 symbols

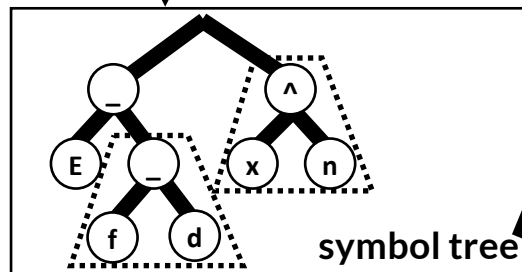
$$E_{f_d} = x^n$$

Precision: 96%  
Recall: 88%

LaTeX

$E_{f_d} = x^n$

parser



E: (start=0, end=1)  
f: (start=3, end=4)...

color command offsets

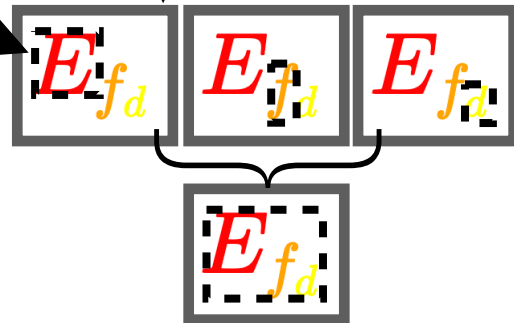
Colorized Equation

$$E_{f_d} = x^n$$

identify token locations

$$E_{f_d} = x^n$$

coalesce



# The construction of ScholarPhi

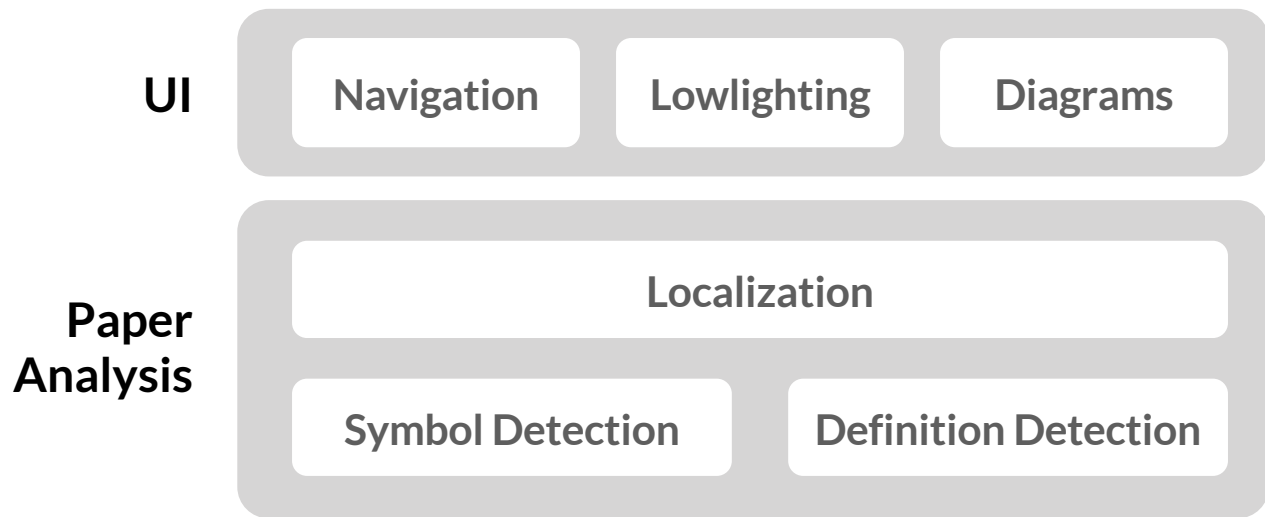
**Major challenge: fine-grained locations of symbols**

**Solution: Link latex code to location in a PDF image**

## **Problems:**

- 1) Difficult to get high accuracy efficiently
- 2) Does not work on PDFs without latex

# The construction of ScholarPhi



**Major challenge: definition recognition**

# The construction of ScholarPhi

**Major challenge: definition recognition**

**Solution: new algorithm for definition detection**



# The construction of ScholarPhi

**Major challenge: definition recognition**

**Solution: new algorithm for definition detection**

## **Problems:**

Difficult to specify what a definition is

Accuracy is still not high enough

# NLP challenge: recognize complex definitions

.. we represent the projected box  $b_p^i$  as a 4-dimensional vector  $\mathbf{d}_{\{i,j\}}^i = [d_{\{t_{\{i,j\}}\}}^i, d_{\{l_{\{i,j\}}\}}^i, d_{\{b_{\{i,j\}}\}}^i, d_{\{r_{\{i,j\}}\}}^i]$ , where  $d_{t^i}$ ,  $d_{l^i}$ ,  $d_{b^i}$ ,  $d_{r^i}$  are the distances between the current pixel location  $(i, j)$  and the top, left, bottom, and right boundaries of  $b_p^i$ , **respectively**. ..

# NLP challenge: recognize referential linkages

.. we represent the projected box  $b_p^i$

as a 4-dimensional vector  $\mathbf{d}_{i,j}^b$

$\mathbf{d}_{i,j}^b = [d_{t_{i,j}}^b, d_{l_{i,j}}^b,$

$d_{b_{i,j}}^b, d_{r_{i,j}}^b]$ , where  $d_{t_{i,j}}^b$

,  $d_{l_{i,j}}^b$ ,  $d_{b_{i,j}}^b$ ,  $d_{r_{i,j}}^b$  are the distances

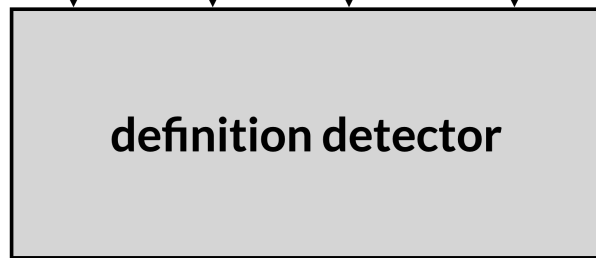
between the current pixel location  $(i, j)$

and the  $top$ ,  $left$ ,  $bottom$ , and  $right$

boundaries of  $b_p^i$ , respectively. ..

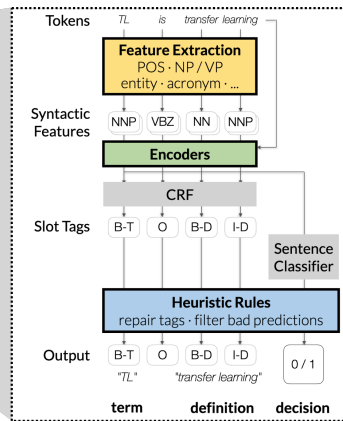
# Heddex/Taddex: State-of-the-art Definition Recognition

*“ TL is transfer learning ”*



**TL**      *transfer learning*  
**term**      **definition**

These models are seeing >70% F1 accuracy on definition prediction.



Kang et al., *EMNLP-SDP '20*



Dongyeop Kang



Andrew Head



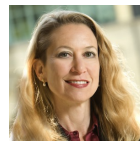
Risham Sirdu



Kyle Lo



Daniel S. Weld



Marti A. Hearst

# Can a LLM Solve These Problems?

GPT-4 is the leading LLM: How well does it do?

# Why GPT-4?

## Comparisons from the Llama 2 Paper

Benchmark (shots)	GPT-3.5	GPT-4	PaLM	PaLM-2-L	LLAMA 2
MMLU (5-shot)	70.0	<b>86.4</b>	69.3	78.3	68.9
TriviaQA (1-shot)	–	–	81.4	<b>86.1</b>	85.0
Natural Questions (1-shot)	–	–	29.3	<b>37.5</b>	33.0
GSM8K (8-shot)	57.1	<b>92.0</b>	56.5	80.7	56.8
HumanEval (0-shot)	48.1	<b>67.0</b>	26.2	–	29.9
BIG-Bench Hard (3-shot)	–	–	52.3	<b>65.7</b>	51.2

Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." *arXiv:2307.09288* (2023).



# GPT-4 is trained on text + images and can do a surprising wide range of tasks

User Answer question I.1.a. Think step-by-step.

## I. Principe de la détection de rayonnement avec un bolomètre

Comme illustré sur la figure 1 un bolomètre est constitué d'un absorbeur qui reçoit le rayonnement que l'on désire détecter. Sa température  $T$ , supposée uniforme, est mesurée à l'aide d'un thermomètre incorporé, constitué d'un matériau conducteur dont la résistance  $R(T)$  varie avec la température  $T$ ; cette variation est caractérisée par le coefficient  $\alpha = \frac{1}{R} \frac{dR}{dT}$ . L'ensemble possède la capacité thermique  $C_{th}$ .

Un barreau, conducteur thermique, homogène, de longueur  $L$ , de section  $S$  et de conductivité thermique  $\lambda$  et sans échanges thermiques latéraux, relie le bolomètre à un thermostat de température  $T_b$  fixe.

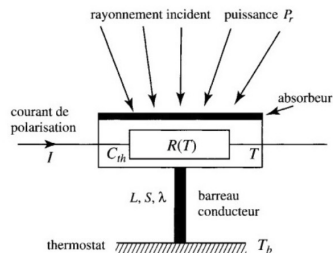


Figure 1 - Schéma d'un bolomètre

I.1. Dans cette première partie, on étudie le comportement du bolomètre en régime stationnaire, à la température de fonctionnement  $T = T_b$ .

I.1.a) Exprimer la température  $T(x)$  en tout point du barreau conducteur repéré par son abscisse  $x$ ; on prendra  $T(0) = T_b$  et  $T(L) = T_b$ .

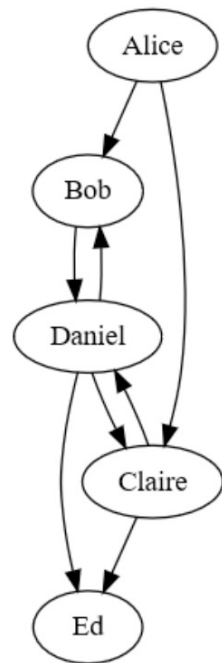
GPT-4 To answer question I.1.a, we need to express the temperature  $T(x)$  in every point of the conductive bar, which is characterized by the abscissa  $x$ . We are given that  $T(0) = T_0$  and  $T(L) = T_b$ .

# GPT-4 is training on text + images and can do a surprising wide range of tasks

I am going to give you a description of a short scene, and I would like you to provide me with a DOT graph representation in a code block where each person in the scene is a node, and whenever two characters interact there is an edge between them:

When first entering the office, Alice greeted Bob who was sitting at his desk and working at the final report. Until the end of the day, Alice never left her cubicle, only briefly talking with Claire during lunch. Most of the afternoon, Daniel and Bob were doing pair programming in the side room, and at the end of the day Dani, Claire and Ed had a meeting.

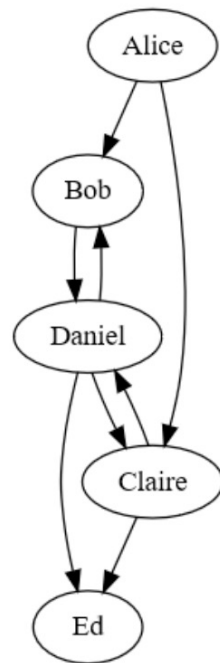
Query



# GPT-4 is training on text + images and can do a surprising wide range of tasks

The above DOT graph represents the scene described in the prompt. Each character is a node, and whenever two characters interact, there is an edge between them. For example, Alice greets Bob when she first enters the office, so there is an edge between Alice and Bob. Similarly, Alice briefly talks with Claire during lunch, so there is an edge between Alice and Claire. Daniel and Bob do pair programming, so there is an edge between them. Finally, Daniel, Claire, and Ed have a meeting at the end of the day, so there are edges between Daniel and Claire, Daniel and Ed, and Claire and Ed.

**Answer**



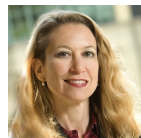
# Does GPT-4 Solve the Definition Recognition Problem?



Anna Martin-  
Boyle



Dongyeop Kang



Marti A. Hearst



HE

For this sentence, what do the labels in the set refer to?

Each character in a sentence is assigned with a unique label from the set  $\{B, M, E, S\}$ , where  $\{B, M, E\}$  indicate Begin, Middle, End of a multi-character word respectively, and S denotes a Single character word.

HE

For this sentence, what do the labels in the set refer to?

Each character in a sentence is assigned with a unique label from the set  $\{B, M, E, S\}$ , where  $\{B, M, E\}$  indicate Begin, Middle, End of a multi-character word respectively, and S denotes a Single character word.



In the given sentence, the labels from the set  $\{B, M, E, S\}$  are used to indicate the segmentation of words within the sentence. Each character in a word is assigned a label according to its position within the word. The labels have the following meanings:

- **B:** Begin - This label indicates the beginning character of a multi-character word.
- **M:** Middle - This label indicates a character that is in the middle of a multi-character word.
- **E:** End - This label indicates the end character of a multi-character word.
- **S:** Single - This label is used for single-character words.

HE

For this sentence, what do the symbols refer to?

We represent the projected box  $b_p^i$  as a 4-dimensional vector  $\mathbf{d}_{i,j} = [d_{t_{i,j}}^i, d_{l_{i,j}}^i, d_{b_{i,j}}^i, d_{r_{i,j}}^i]$ , where  $d_t^i, d_l^i, d_b^i, d_r^i$  are the distances between the current pixel location  $(i, j)$  and the top, left, bottom, and right boundaries of  $b_p^i$ , respectively. ..



HE

For this sentence, what do the symbols refer to?

We represent the projected box  $b_p$  as a 4-dimensional vector  $\mathbf{d}_{i,j} = [d_{t_{i,j}}, d_{l_{i,j}}, d_{b_{i,j}}, d_{r_{i,j}}]$ , where  $d_t$ ,  $d_l$ ,  $d_b$ ,  $d_r$  are the distances between the current pixel location  $(i, j)$  and the top, left, bottom, and right boundaries of  $b_p$ , respectively. ..



In the given sentence, the symbols refer to various components and values related to representing a projected box. Here's a breakdown of what each symbol refers to:

- **$b_p$** : This refers to the projected box labeled as " $b_p$ ." It's a box or rectangle that has been projected onto some context or space.
- **$\mathbf{d}_{i,j}$** : This represents a 4-dimensional vector labeled as " $\mathbf{d}_{i,j}$ ." It's a vector used to store information about the projected box at the pixel location  $(i, j)$ .
- **$d_{t_{i,j}}$** : This refers to the distance labeled as " $d_{t_{i,j}}$ ." It represents the distance between the current pixel location  $(i, j)$  and the top boundary of the projected box.
- **$d_{l_{i,j}}$** : This refers to the distance labeled as " $d_{l_{i,j}}$ ." It represents the distance between the current pixel location  $(i, j)$  and the left boundary of the projected box.
- **$d_{b_{i,j}}$** : This refers to the distance labeled as " $d_{b_{i,j}}$ ." It represents the distance between the current pixel location  $(i, j)$  and the bottom boundary of the projected box.
- **$d_{r_{i,j}}$** : This refers to the distance labeled as " $d_{r_{i,j}}$ ." It represents the distance

# Does GPT-4 Solve the Definition Recognition Problem?

It does well, but our specialized algorithm still performs (a bit) better.

	F1
TADDEX on terms	81.5
GPT-4 on terms	79.7
TADDEX on definitions	73.6
GPT-4 on definitions	70.4

# Does GPT-4 Solve The Document Recognition Problem?

Can it recognize equation structure and location?

SHANNON ZEJIANG SHEN

I am a first year PhD Student at MIT **CSAIL**,  
working at the intersection between **NLP** and **HCI**.



Research

Publications

Contact



# VILA

Improved Structured Content Extraction from Scientific PDFs  
Using Visual Layout Groups

@ACL TACL '22.

# LAYOUTPARSER

A Unified Toolkit for Deep Learning Based Document Image Analysis

@ICDAR '21.

Shannon Shen

## SHANNON ZEJIANG SHEN

I am a first year PhD Student at MIT **CSAIL**,  
working at the intersection between **NLP** and **HCI**.



Research

Publications

Contact

# Layout Analysis – in 4 Lines of Code

Transform document image analysis pipelines with the full power of Deep Learning.

```
 pip install layoutparser
```

## What is Layout Parser?

A Unified Toolkit for Deep Learning Based Document Image Analysis

### LAYOUT DETECTION

Deep Learning Models

Pre-trained Model Zoo

### POST-PROCESSING

Layout Data structure

Visualization

## Accurate Layout Detection with a Simple and Clean Interface

With the help of state-of-the-art deep learning models, Layout Parser enables extracting complicated document structures using only several lines of code. This method is also more robust and generalizable as no sophisticated rules are involved in this process.

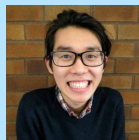
```
import layoutparser as lp
image = cv2.imread(...)
```



# PAPERMAGE

A unified Toolkit for Processing, Representing, and  
Manipulating Visually-Rich Scientific Documents

(under review)



Kyle Lo



Shannon Shen



Daniel S. Weld

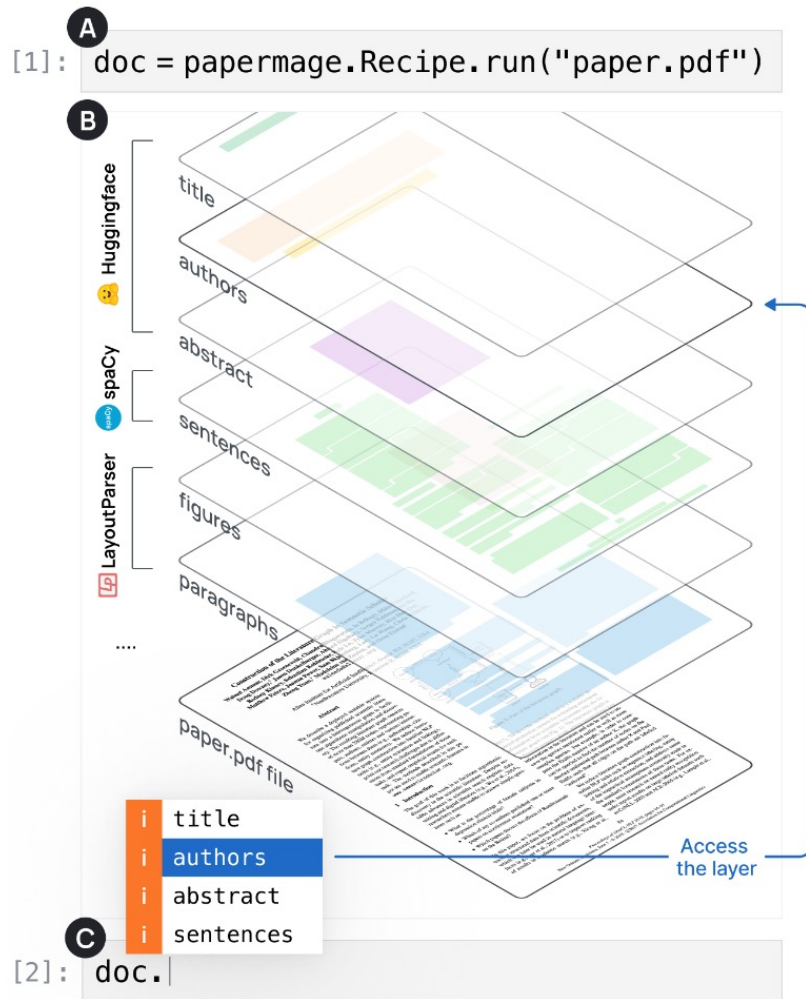


Luca  
Soldaini

And many  
other authors

PaperMage allows for the composition of different machine learning and layout analysis models

Access the various layers via python





```
>>> doc.paragraphs[0]
```

```
>>> doc.paragraphs[0].sentences[2]
```

```
or
```

```
>>> doc.sentences[2]
```

```
>>> doc.tokens[169:173]
```

```
>>> doc.figures[0]
```

```
>>> q = Box(l,t,w,h, page=0)
```

```
>>> stoks = doc.find(q, layer="tokens")
```

```
>>> [token.text for token in stoks]
```

#### ABSTRACT

Crowdsourcing provides a scalable and efficient way to construct labeled datasets for training machine learning systems. However, creating comprehensive label guidelines for crowdworkers is often prohibitive even for seemingly simple concepts. Incomplete or ambiguous label guidelines can then result in differing interpretations of concepts and inconsistent labels. Existing approaches for improving label quality, such as worker screening or detection of poor work, are ineffective for this problem and can lead to rejection of honest work and a missed opportunity to capture rich interpretations about data. We introduce *Revolt*, a collaborative approach that brings ideas from expert annotation workflows to crowd-based labeling. Revolt eliminates the burden of creating detailed label guidelines by harnessing crowd disagreements to identify ambiguous concepts and create rich structures (groups of semantically related items) for post-hoc label decisions. Experiments comparing Revolt to traditional crowdsourced labeling show that Revolt produces high quality labels without requiring label guidelines in turn for an increase in monetary cost. This up front cost, however, is mitigated by Revolt's ability to produce reusable structures that can accommodate a variety of label boundaries without requiring new data to be collected. Further comparisons of Revolt's collaborative and non-collaborative variants show that collaboration reaches higher label accuracy with lower monetary cost.

#### ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

#### Author Keywords

crowdsourcing; machine learning; collaboration; real-time

#### INTRODUCTION

From conversational assistants on mobile devices, to facial

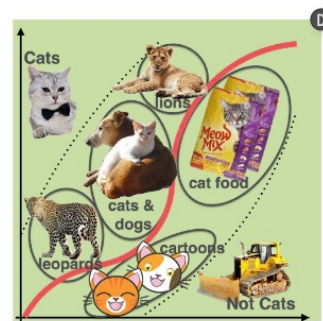


Figure 1. Revolt creates labels for unanimously labeled “certain” items (e.g., cats and not cats), and surfaces categories of “uncertain” items enriched with crowd feedback (e.g., cats and dogs and cartoon cats in the dotted middle region are annotated with crowd explanations). Rich structures allow label requesters to better understand concepts in the data and make post-hoc decisions on label boundaries (e.g., assigning cats and dogs to the cats label and cartoon cats to the not cats label) rather than providing crowd-workers with a priori label guidelines.

learned models that must be trained on representative datasets labeled according to target concepts (e.g., speech labeled by their intended commands, faces labeled in images, emails labeled as spam or not spam).

Techniques for collecting labeled data include recruiting experts for manual annotation [51], extracting relations from readily available sources (e.g., identifying bodies of text in parallel online translations [46, 13]), and automatically generating labels based on user behaviors (e.g., using dwell time to implicitly mark search result relevance [2]). Recently, many practitioners have also turned to crowdsourcing for creating labeled datasets at low cost [49]. Successful crowd-

# PaperMage supports structured algorithm comparison

Structure Category	GROBID <sub>CRF</sub>			GROBID <sub>NN</sub>			RoBERTa			I-VILA		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Abstract	81.9	89.1	85.3	85.3	89.8	87.5	89.2	93.7	91.4	97.4	98.3	97.8
Author	55.2	42.6	48.1	75.1	14.0	23.6	87.5	73.5	79.9	65.5	96.9	78.2
Bibliography	96.5	98.6	97.5	95.5	97.6	96.5	93.6	93.3	93.5	99.7	98.2	99.0
Caption	70.3	70.0	70.2	70.2	69.7	70.0	80.0	77.3	78.6	93.1	89.6	91.3
Equation	71.1	85.3	77.6	71.1	85.3	77.6	55.0	85.7	67.0	90.7	94.2	92.4
Figure	0.0	0.0	0.0	0.0	0.0	0.0	88.9	82.3	85.4	99.8	96.8	98.3
Footer	0.0	0.0	0.0	0.0	0.0	0.0	56.1	59.9	57.9	96.8	78.1	86.5
Footnote	0.0	0.0	0.0	0.0	0.0	0.0	59.8	44.3	50.9	80.2	93.5	86.3
Header	0.0	0.0	0.0	0.0	0.0	0.0	40.5	84.3	54.7	92.9	99.1	95.9
Keywords	0.0	0.0	0.0	0.0	0.0	0.0	93.8	97.1	95.4	96.9	99.4	98.1
List	0.0	0.0	0.0	0.0	0.0	0.0	61.9	63.8	62.9	76.7	82.4	79.4
Paragraph	94.5	89.8	92.1	94.4	89.9	92.1	93.5	93.0	93.3	98.7	97.9	98.3
Section	83.0	79.4	81.1	83.0	79.4	81.1	67.7	82.7	74.4	96.2	91.6	93.9
Table	97.3	58.6	73.2	97.9	58.6	73.3	94.7	71.8	81.7	96.1	94.9	95.5
Title	0.0	0.0	0.0	0.0	0.0	0.0	76.3	96.7	85.3	98.7	99.9	99.3
<b>Macro Avg</b> (Full S2-VL)	40.6	38.3	39.1	42.0	36.5	37.6	75.9	80.0	76.8	92.0	94.1	92.7
<b>Macro Avg</b> (Grobid Subset)	81.2	76.7	78.9	84.1	73.0	78.2	82.6	83.9	83.2	92.2	95.2	93.7

## GPT-4?

# Does GPT-4 Solve The Document Recognition Problem?

Can it recognize equation structure and location?

SHANNON ZEJIANG SHEN

I am a first year PhD Student at MIT **CSAIL**,  
working at the intersection between **NLP** and **HCI**.



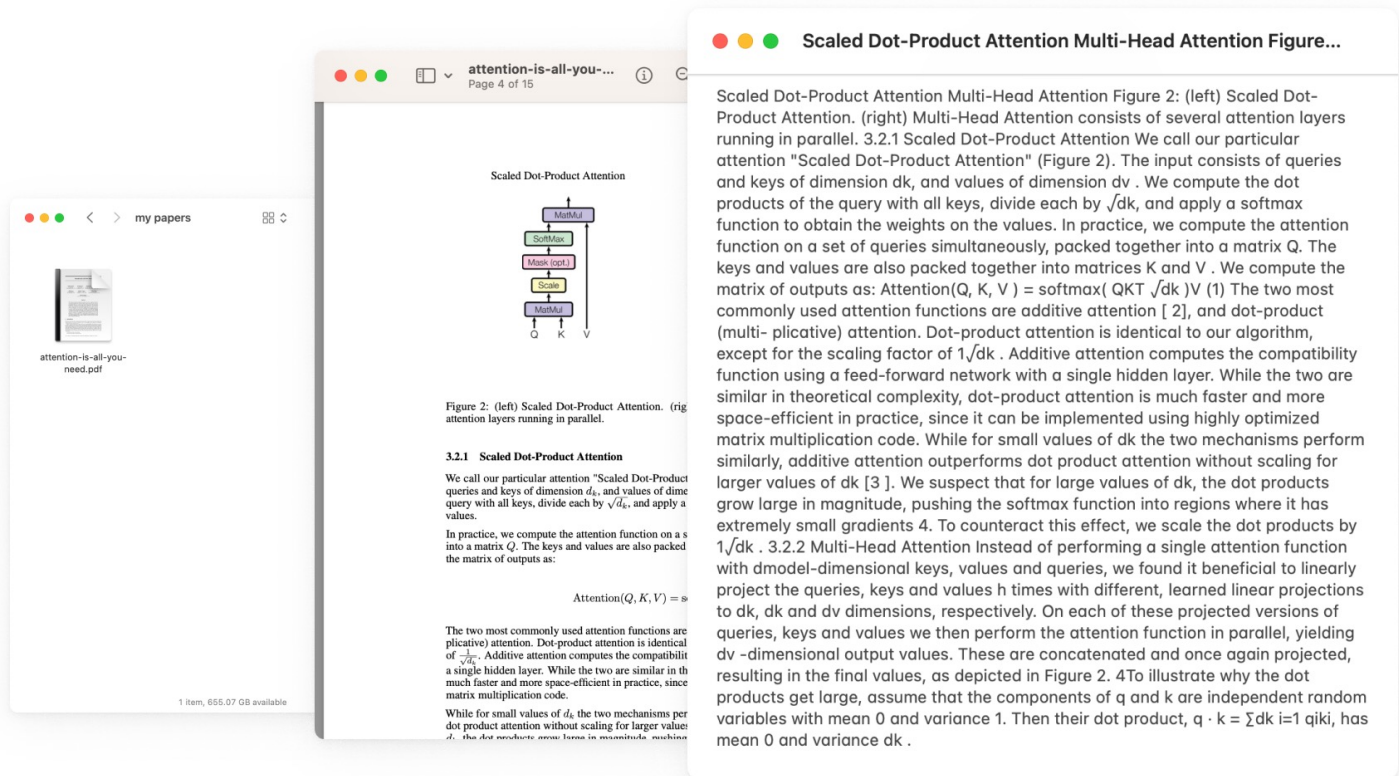
Research

Publications

Contact



# A typical PDF Parsing Pipeline



# Unstructured PDF text cannot faithfully represent the equation

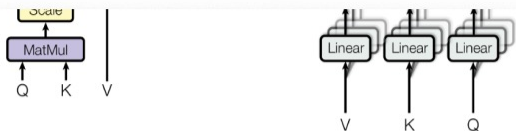


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

## 3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ . We compute the dot products of the query with all keys, divide each by  $\sqrt{d_k}$ , and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix  $Q$ . The keys and values are also packed together into matrices  $K$  and  $V$ . We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of  $\frac{1}{\sqrt{d_k}}$ . Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

While for small values of  $d_k$  the two mechanisms perform similarly, additive attention outperforms dot product attention without scaling for larger values of  $d_k$  [3]. We suspect that for large values of  $d_k$ , the dot products grow large in magnitude, pushing the softmax function into regions where it has extremely small gradients<sup>4</sup>. To counteract this effect, we scale the dot products by  $\frac{1}{\sqrt{d_k}}$ .

## 3.2.2 Multi-Head Attention

...The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ . We compute the dot products of the query with all keys, divide each by  $\sqrt{d_k}$ , and apply a softmax function to obtain the weights on the values. In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix  $Q$ . The keys and values are also packed together into matrices  $K$  and  $V$ . We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T \sqrt{d_k})V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of  $1/\sqrt{d_k}$ . Additive attention computes the compatibility function...



# GPT-4 can analyze pdf text output ...

... on the values. In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q. The keys and values are also packed together into matrices K and V .We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multi- plicative) attention. Dot-product attention is identical to our algorithm, except for the scal...

Unstructured PDF Text

M

Can you detect the equations from the PDF text and convert it into latex?



$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

M

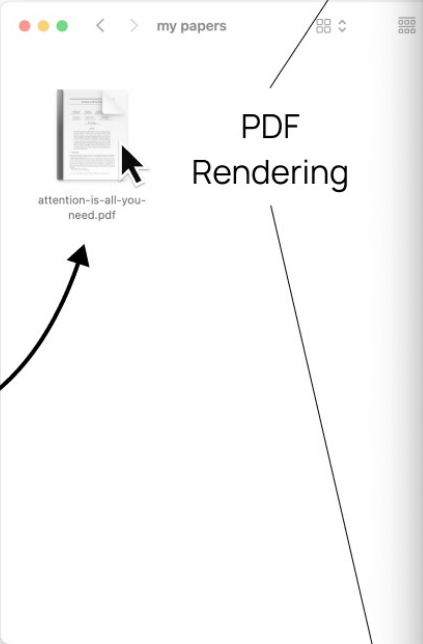
**But it is not correct.**

Here is the original equation and the subscript is wrong:

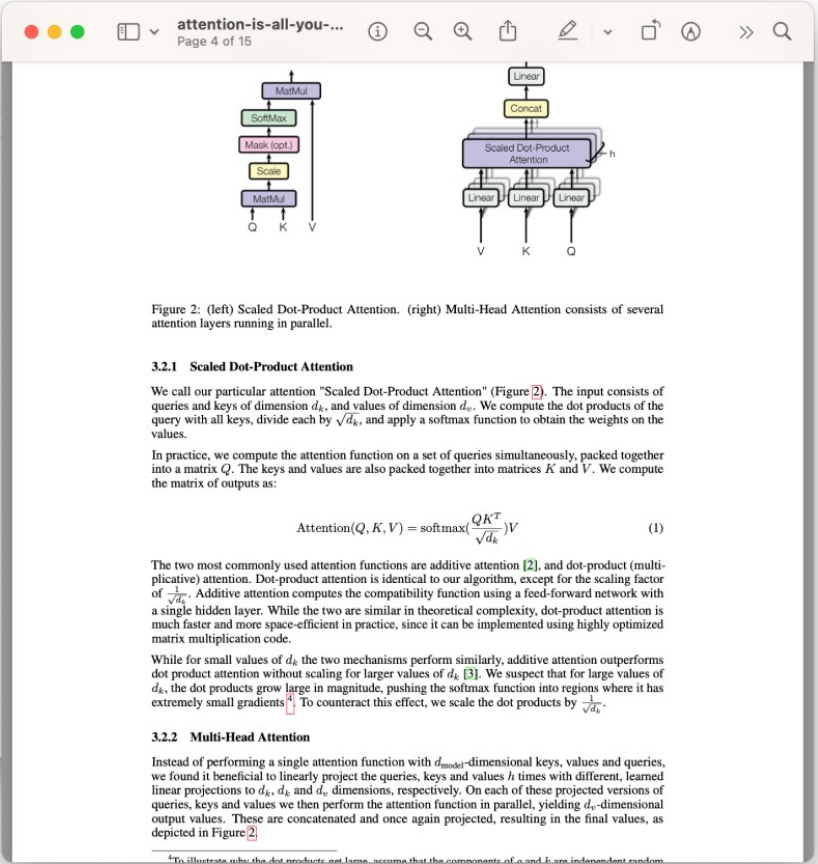
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

... but the lack of structure  
can cause errors in downstream tasks

What's inside  
a PDF File?



PDF  
Rendering



The PDF File

Rendered PDF



# What is inside a PDF File?

## PDF Code → PDF Objects

Describes how to display text and other objects on a page.

```
BT /F61 9.9626 Tf 1.02 0 0 1 108 509.981 Tm  
[(Figure)-341(2:)-495((left))-340(Scaled)-341(Dot-  
Product)-341(Attention.)-590((right))-341(Multi-  
Head)-341(At)1(tention)-341(consists)-341(of)-340(  
se)24(v)15(eral)]1 0 0 1 108 499.072 Tm  
[(attention)-250(layers)-250(running)-250(in)-250(  
parallel.))]0 g 0 G
```

# PDF Code corresponding to the highlighted heading

## PDF Page Render

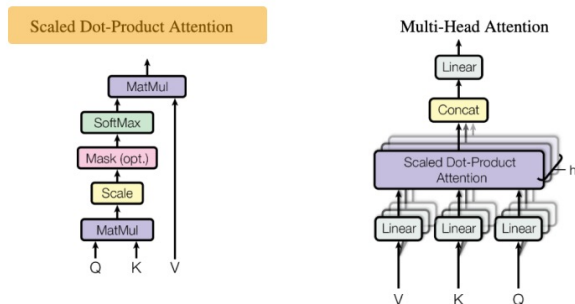


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

### 3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ . We compute the dot products of the query with all keys, divide each by  $\sqrt{d_k}$ , and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix  $Q$ . The keys and values are also packed together into matrices  $K$  and  $V$ . We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of  $\frac{1}{\sqrt{d_k}}$ . Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

## PDF Code

...

```
BT
/F61 9.9626 Tf
147.783 713.245 Td
[(Scaled)-250(Dot-
Product)-250(Attention)]TJ
ET
```

Begin Text Chunk

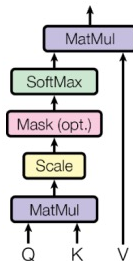
Use Font 61 + 9.9 pt

Adjust the position

Display the text

End Text Chunk

Scaled Dot-Product Attention



Multi-Head Attention

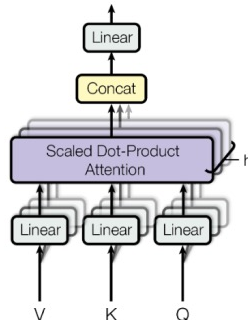


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

### 3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ . We compute the dot products of the query with all keys, divide each by  $\sqrt{d_k}$ , and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix  $Q$ . The keys and values are also packed together into matrices  $K$  and  $V$ . We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of  $\frac{1}{\sqrt{d_k}}$ . Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

```
/F8 9.9626 Tf 111.97 -34.666 Td
```

```
[(A)28(tten)28(tion())]/F11 9.9626 Tf
```

```
45.662 0 Td [(Q,)-167(K)-16(,)-166(V)]/F8
```

```
9.9626 Tf 33.378 0 Td
```

```
[() )-278(=)-277(s)-1(of)1(tm)-1(ax())]/F11
```

```
9.9626 Tf 56.599 6.74 Td [(QK)]/F10 6.9738
```

```
Tf 17.05 3.615 Td [(T)]ET q 1 0 0 1
```

```
355.609 343.199 cm []0 d 0 J 0.398 w 0 0 m
```

```
23.326 0 l S Q BT /F14 9.9626 Tf 358.077
```

```
341.007 Td [(v)]ET q 1 0 0 1 366.379
```

```
341.206 cm []0 d 0 J 0.398 w 0 0 m 10.087
```

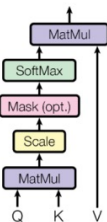
```
0 l S Q BT /F11 9.9626 Tf 366.379 333.264
```

```
Td [(d)]/F10 6.9738 Tf 5.186 -1.494 Td [(...
```

# PDF Code corresponding to the highlighted equation

## PDF Page Render

Scaled Dot-Product Attention



Multi-Head Attention

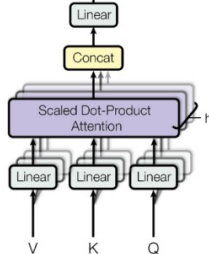


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

### 3.2.1 Scaled Dot-Product Attention

We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of queries and keys of dimension  $d_k$ , and values of dimension  $d_v$ . We compute the dot products of the query with all keys, divide each by  $\sqrt{d_k}$ , and apply a softmax function to obtain the weights on the values.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix  $Q$ . The keys and values are also packed together into matrices  $K$  and  $V$ . We compute the matrix of outputs as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

The two most commonly used attention functions are additive attention [2], and dot-product (multiplicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor of  $-\frac{1}{\sqrt{d_k}}$ . Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice, since it can be implemented using highly optimized matrix multiplication code.

## PDF Code

```
/F8 9.9626 Tf 111.97 -34.666 Td
[(A)28(tten)28(tion())]/F11 9.9626 Tf
45.662 0 Td [(Q,)-167(K)-16(,)-166(V)]/F8
9.9626 Tf 33.378 0 Td
[( )-278(=)-277(s)-1(of)1(tm)-1(ax())]/F11
9.9626 Tf 56.599 6.74 Td [(QK)]/F10 6.9738
Tf 17.05 3.615 Td [(T)]ET q 1 0 0 1
355.609 343.199 cm []0 d 0 J 0.398 w 0 0 m
23.326 0 l S Q BT /F14 9.9626 Tf 358.077
341.007 Td [(√)]ET q 1 0 0 1 366.379
341.206 cm []0 d 0 J 0.398 w 0 0 m 10.087
0 l S Q BT /F11 9.9626 Tf 366.379 333.264
Td [(d)]/F10 6.9738 Tf 5.186 -1.494 Td [(...
```

# It turns out that GPT-4 can analyze PDF code!

```
/F8 9.9626 Tf 111.97 -34.666  
Td [(A)28(tten)28(tion())]/F11  
9.9626 Tf 45.662 0 Td  
[(Q,)-167(K)-16(,)-166(V)]/F8  
9.9626 Tf 33.378 0 Td  
[( )]-278(=)-277(s)-1(of)1(tm)  
-1(ax())]/F11 9.9626 Tf 56.599  
6.74 Td [(QK)]/F10 6.9738 Tf  
17.05 3.615 Td [(T)]ET q 1 0  
0 1 355.609 343.199 cm []0 d  
0 J 0.398 w 0 0 m 23.326 0 l  
S Q BT /F14 9.9626 Tf 358.07...
```

The source PDF Code

M

Can you detect the equations from the PDF Code and convert it into latex?



$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

M

**Great! It is correct!**

Here is the original equation for reference.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



**It gets the subscript correct that it missed from pdf text alone**

# How does this work? Let's dive deeper into the PDF code

```
[(eigenvalue)-307(of)-308(the)-307(normalized)-307(ad  
jacency)-308(matrix)]TJ ET q 1 0 0 1 301.539 677.183  
cm []0 d 0 J 0.438 w 0 0 m 8.182 0 l S Q BT /F36  
10.9091 Tf 301.539 668.194 Td [(A)]TJ/F98 10.9589 Tf  
11.551 0 Td [(and)]TJ/F36 10.9091 Tf 21.921 0 Td  
[(γ)]TJ/F35 7.9701 Tf 5.648 -1.636 Td [(1)]TJ/F34  
10.9091 Tf 8.922 1.636 Td [(=)-384(1)]TJ/F98 10.9589  
Tf 18.129 0 Td  
[(.)-482(Accor)18(ding)-307(to)-308(Cheeger)-74('s)-3  
07(inequality)]TJ -299.672 -12.951 Td
```

M

There is an inline equation in this pdf code:

1.  prints the letter  $\gamma$  with a regular font size.
2.  prints the number 1 with a **smaller** font size, and the text is shifted downside. Thus 1 is a subscript. The code is ``_{1}``.

Therefore, the equation latex is ``$ \gamma_{1}=1$``.

B has smaller font than A; D shows the position shifting down

# Correct Parsing

---

Original PDF

$$\Pr[M(D) \in S] \leq \exp(\epsilon) \Pr[M(D') \in S] + \delta.$$

---

PDF Code Input  
(3800 chars)

```
.../F24 9.9626 Tf -147.876 -18.267 Td [(Pr[])]/
F27 9.9626 Tf 13.449 0 Td [(M)]/F24 9.9626 Tf
10.752 0 Td [(())]/F27 9.9626 Tf 3.874 0 Td
[(D)]/F24 9.9626 Tf 8.525 0 Td [(())]/F30
9.9626 Tf 6.642 0 Td [(ε)]/F27 9.9626 Tf 9.409
0 Td [(S)]/F24 9.9626 Tf 6.683 0 Td [(())]/F30...
```

→ GPT-4 → Latex:

$$\Pr[M(D) \in S] \leq \exp(\epsilon) \Pr[M(D') \in S] + \delta.$$

---

PDF Text Input  
(717 chars)

```
...datasets  $D, D'$  and for any  $S \subseteq \text{range}(M)$ ,
 $\Pr[M(D) \in S] \leq \exp(\epsilon) \Pr[M(D') \in S] + \delta.$ 
We say two datasets  $D, D' \in X$  are neighboring
if they differ on at most an individual's
participation. Two additional steps are added
to the FL algorithm to ensure a DP guar- ...
```

→ GPT-4 → Latex:

$$\Pr[M(D) \in S] \leq \exp(\epsilon) \Pr[M(D') \in S] + \delta$$

---

# Failure Cases

## Complex Subscript Structure

---

Original PDF

$$\mathcal{E}(f_{\text{pop}}^*) \leq \tilde{O}\left(\alpha/\rho^2 \left\lfloor \frac{k}{2} \right\rfloor\right).$$

---

PDF Code → GPT-4 → Latex

$$E(f_{\text{pop}}^*) \leq \tilde{O}\left(\frac{\alpha}{\rho^2} \left\lfloor \frac{k}{2} \right\rfloor\right)$$

---

PDF Text → GPT-4 → Latex

$$E(f_{\text{pop}}^*) \leq \tilde{O}\left(\frac{\alpha}{\rho^2}\right)^{\frac{bk}{2c}}$$

---



# Failure Cases

## Complex and Long Equations

Original PDF

$$\frac{1}{\lambda} \cdot \frac{C - Npq}{N(1-p) + 1} \leq \mathbb{E}[U_k] \leq \frac{C}{\lambda(N-C)}.$$

PDF Code → GPT-4 → Latex

$$\overset{1}{\lambda} \cdot C - Npq \quad \overset{2}{N(1-p) + 1} \leq E[U_k] \leq C \quad \overset{3}{\lambda(N-C)}$$

PDF Text → GPT-4 → Latex

$$\frac{1}{\lambda \cdot C} - \frac{Npq}{N(1-p) + 1} \leq E[U_k] \leq \frac{C}{\lambda(N-C)}$$

# Our Prompts

```
[1]: equation_detection_from_pdf_code(pdf_code=pdf_code, llm=gpt4)
```

```
system    You are a helpful assistant.

Here is a piece of pdf code:

...

[[eigenvalue)-307(of)-308(the)-307(normalized)-307(adjacency)-308(matrix)]TJ
ET
q
1 0 0 1 301.539 677.183 cm
[]0 d 0 J 0.438 w 0 0 m 8.182 0 l S
Q
BT
/F36 10.9091 Tf 301.539 668.194 Td [(A)]TJ/F98 10.9589 Tf 11.551 0 Td [(and)]TJ/F36 10.9091 Tf 21.921 0 Td [(y)]TJ/F35 7.970
1 Tf 5.648 -1.636 Td [(1)]TJ/F34 10.9091 Tf 8.922 1.636 Td [(=)-384(1)]TJ/F98 10.9589 Tf 18.129 0 Td [(.)-482(Accor)18(ding)-30
7(to)-308(Cheeger)-74(s)-307(inequality)]TJ -299.672 -12.951 Td
...

Can you extract the math equation or symbols in this pdf code and convert it into latex?

Yes, I can help. Let's do it step by step:

1. There is an inline equation in "ET q 1 0 0 1 301.539 677.183 cm []0 d 0 J 0.438 w 0 0 m 8.182 0 l S Q BT /F36 10.9091 Tf 301.
539 668.194 Td [(A)]":
- 'q 1 0 0 1 301.539 677.183 cm []0 d 0 J 0.438 w 0 0 m 8.182 0 l S Q' draws a small dash at the top.
- The equation translates to '$\bar{A}$' in latex.

2. There is an inline equation in "[[(y)]TJ/F35 7.9701 Tf 5.648 -1.636 Td [(1)]TJ/F34 10.9091 Tf 8.922 1.636 Td [(=)-384(1)]TJ":
- '/F35 7.9701 Tf 5.648 -1.636 Td [(1)]' prints 1 with a smaller font size and the position is shifted downside. Thus 1 is a subscr
pt. The code is '$_{1}$'.
- The equation translates to '$\gamma_1=1$' in latex.

The final equations are:
- '$\bar{A}$'
- '$\gamma_1=1$'

Here is another piece of pdf code:
```

## In-context learning examples

*We manually write the prompts to teach GPT-4 how to parse the equations from the PDF Code.*

# Does GPT-4 Solve The Document Recognition Problem?

Stay tuned ... we are working on getting detailed results

SHANNON ZEJIANG SHEN

I am a first year PhD Student at MIT **CSAIL**,  
working at the intersection between **NLP** and **HCI**.



Research

Publications

Contact



# To consider: In light of the capabilities of LLMs, what is the role of classic representations?

For NLP: do we need  
syntax and parse  
trees?

For document  
analysis: do we need  
layout structure?

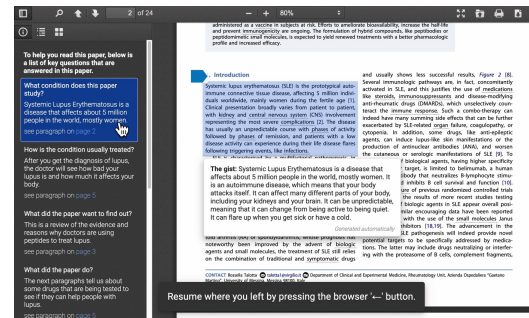
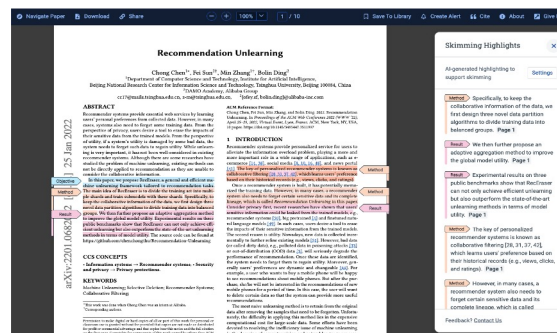
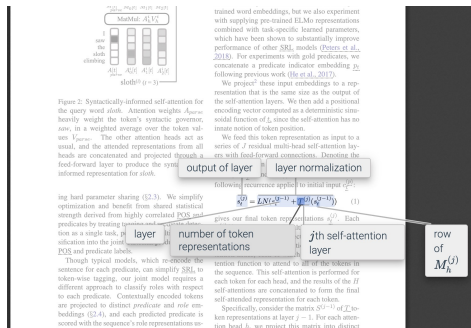
# In Summary

## NLP + HCI + Document Analysis can Improve the Understandability of Scientific Papers

ScholarPhi

Scim

PaperPlain



# Scientific paper interactivity offers great research opportunities for the ICDAR community!

# Bringing Scientific Papers to Life

## Thank you!

Marti Hearst  
UC Berkeley

In collaboration with the Allen Institute of AI

Berkeley SCHOOL OF  
INFORMATION

