LISA

Linguistically-Informed Self-Attention for Semantic Role Labeling





Emma Strubell[1] Patrick Verga[1] Daniel Andor[2] David Weiss[2] Andrew McCallum[1]

[1] UMassAmherst College of Information and Computer Sciences

[2] Google AI

Thanks for the introduction. Today I'm excited to present LISA, a new method for multi-task learning where supervision for transfer between tasks is done through the attention mechanism, yielding new SOTA results on SRL.

This is joint work with my labmate Pat Verga, collaborators Daniel Andor and David Weiss at Google, and my advisor Andrew McCallum.
--

which unifies highly accurate neural network models with robust linguistic structure in order to perform fast, accurate and robust NLP.

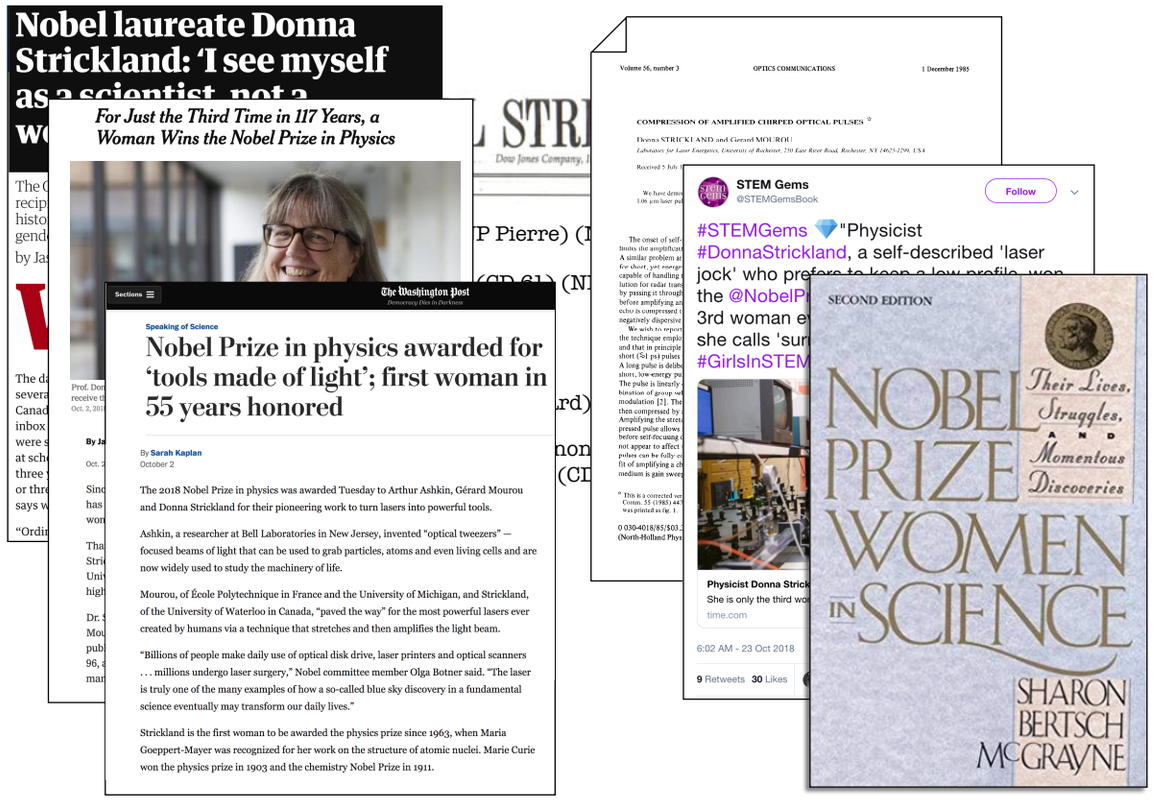
multi-task learning where supervision xfer done through attention mechanism

---

I'm going to demonstrate this in the midst of semantic role labeling and syntactic dependency parsing, where we train a model to predict 4 related tasks with just one pass over the sentence, and achieve state-of-the art SRL F1, with particularly substantial gains when evaluating on out-of-domain text.

on linguistically-informed self-attention for semantic role labeling. By training a single deep neural network model to predict four related tasks and modifying self-attention to attend to syntactic heads, we achieve state-of-the-art results on PropBank SRL, with particularly substantial gains when evaluating on out-of-domain text. I'm going to

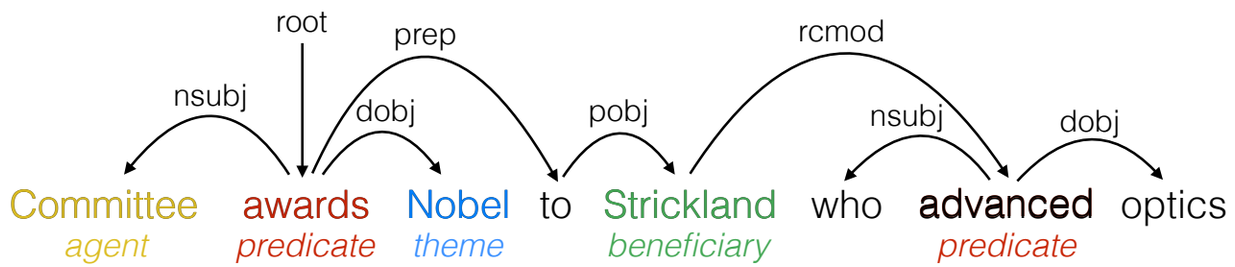
Want fast, accurate, robust NLP



Over the past 10 years or so NLP research has made so much progress that it's no longer just an object of study

1) but, as Gideon mentioned in his keynote yesterday, something that practitioners want to run at massive scale in order to extract meaning from text. So they want these techniques to be computationally efficient, while also obtaining high accuracy, not just on news articles,
2) but also across different domains. The model I'm going to present unifies deep neural network models with linguistic structure in order to perform fast, accurate and robust NLP.
   In this work we're going to focus on the extraction task of semantic role labeling,
3) which I'm going to briefly introduce now.

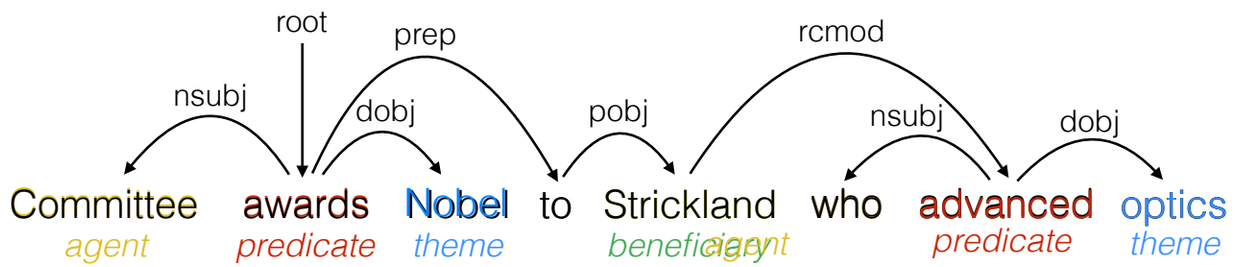
SRL: Who did what to whom?



The task of semantic role labeling is typically cast as extracting "who" did "what" to "whom."
Basic models for this can be built by designing rules over parse trees, but this doesn't scale well across diverse sentence structures.
Instead, we typically directly label the spans corresponding to predicates and their arguments in the sentence.

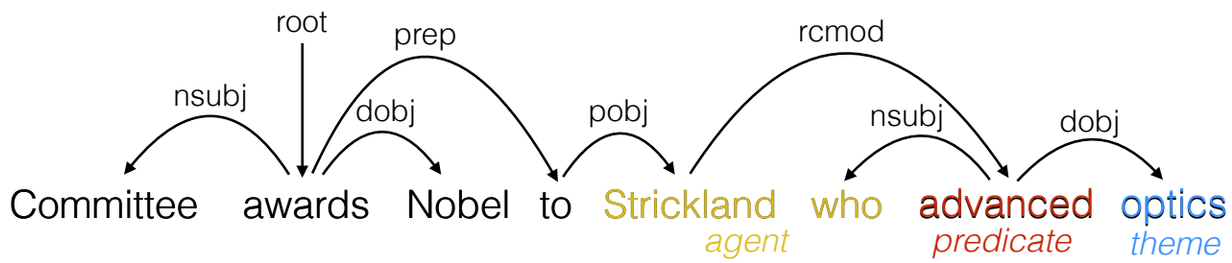
1. For example, in this sentence we would identify the predicates "awards" and "advanced". We can then extract the arguments with respect to each predicate.
2. So with respect to the predicate awards,
3. The "agent" argument, or the one doing the awarding, is "committee", its "theme" or the thing being awarded is the "Nobel", and the "beneficiary", the one receiving the award, is "Strickland.
4. That is the labeling for the predicate "awards"

SRL: Who did what to whom?



1. Similarly, for the predicate "advanced,"
2. the agent is "Strickland"
3. and the theme is "optics"

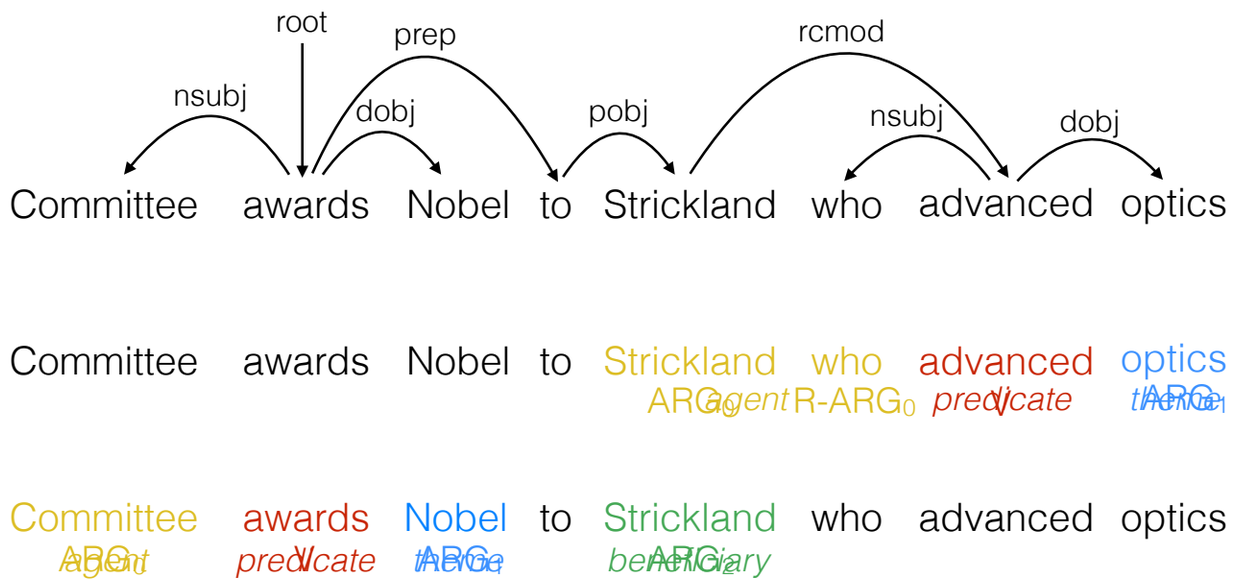
SRL: Who did what to whom?



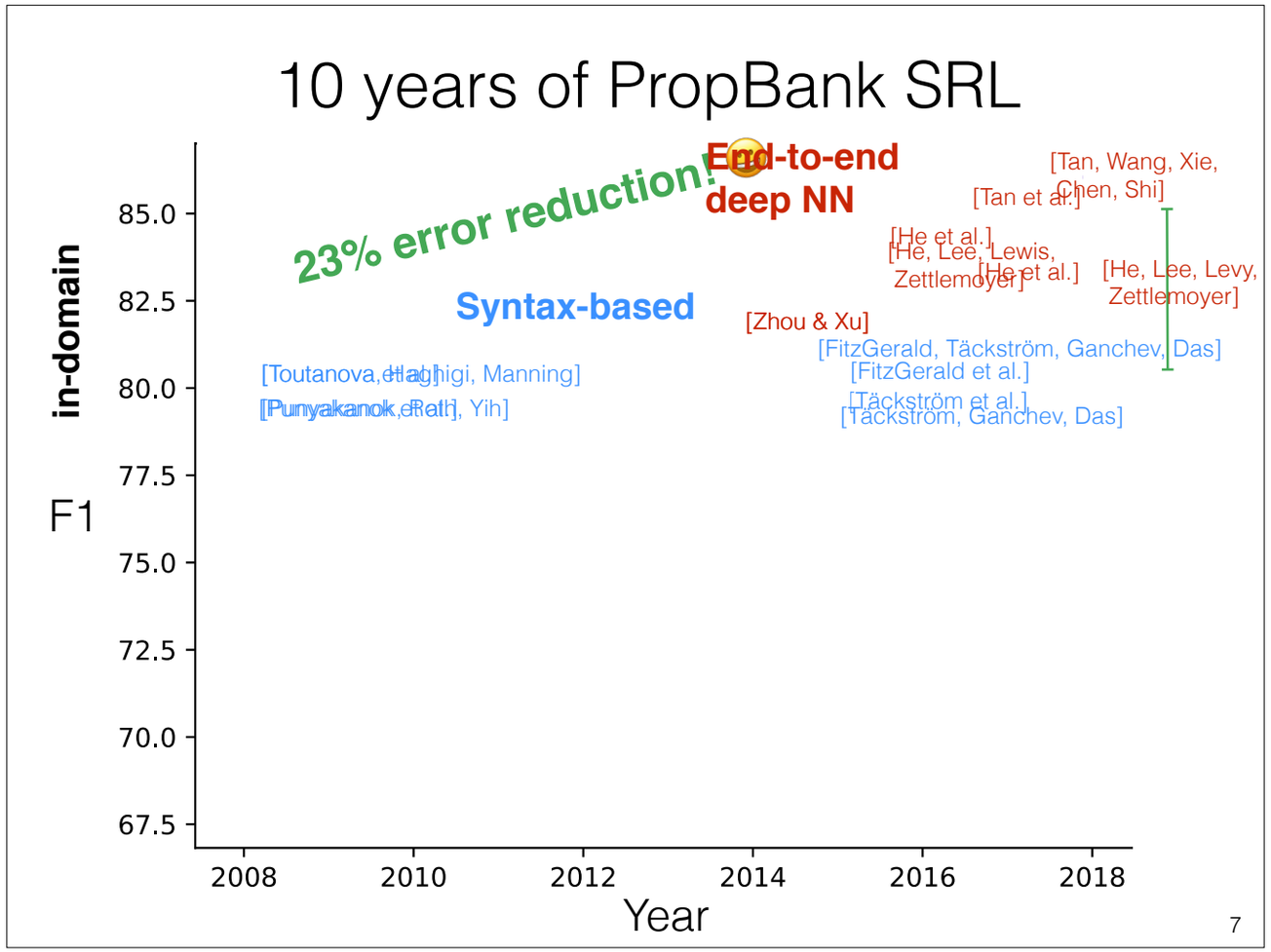
Committee awards Nobel to Strickland who advanced optics
agent *predicate* *theme* *beneficiary*

1. These are the two semantic role labelings of the sentence, with respect to the two predicates.

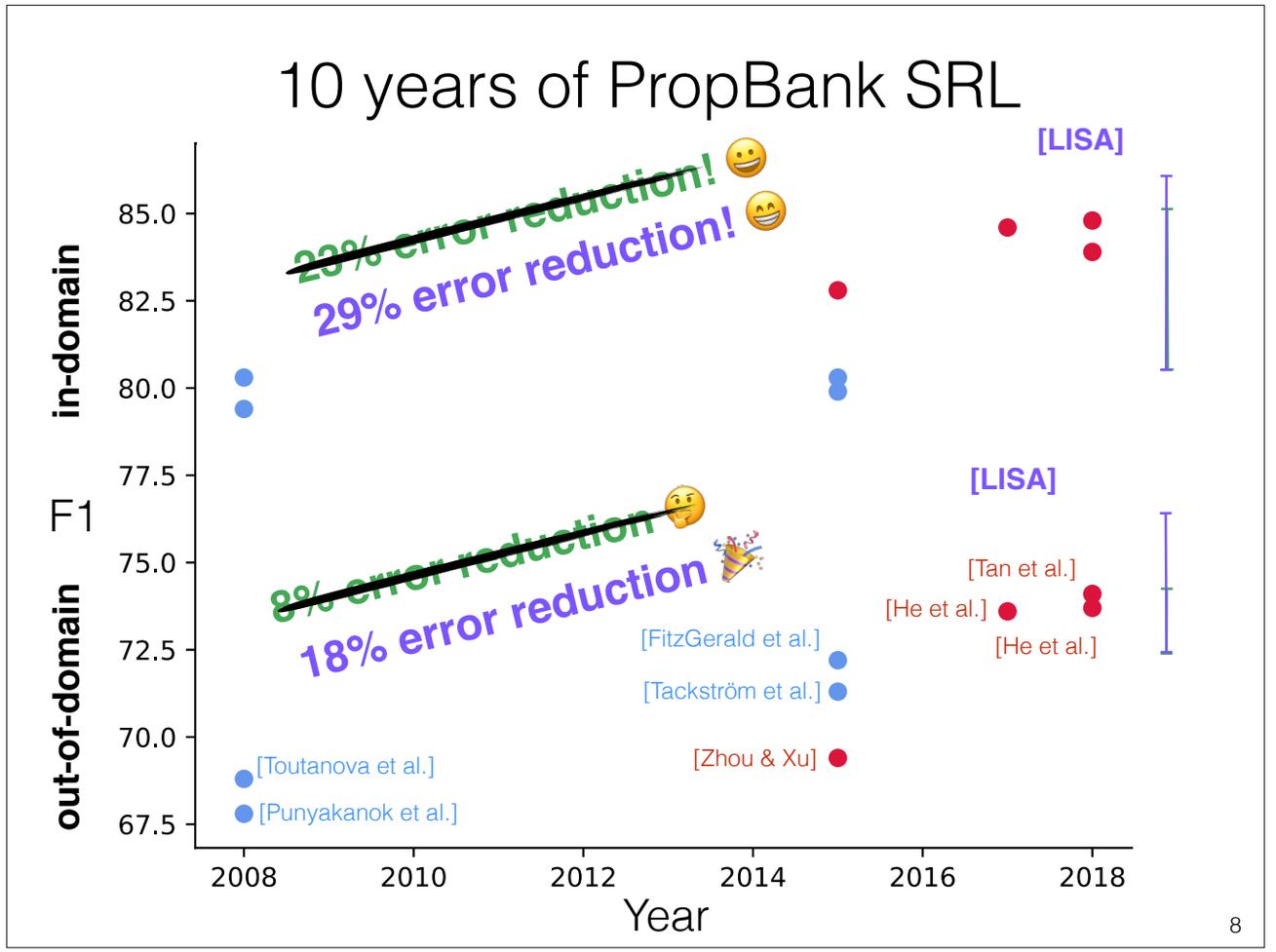
PropBank SRL: Who did what to whom?



1. We're specifically using the PropBank SRL framework, which assigns more general argument labels shared across all predicates. So that's what this looks like in practice.
2. So we want to extract semantics, but understandably this task is deeply intertwined with syntax,
3. And historically,



SRL models have relied heavily on syntax. Looking at the last 10 years of SRL on the CoNLL-2005 shared task,
1) Early SRL models relied heavily on syntax, typically using a combination of syntactic features on the input and syntactic constraints during inference.
2) More recently, there's been a trend where end-to-end deep NNs which use NO syntax are surpassing their linguistically-informed counterparts, not just for semantic role labeling but this is a trend across many NLP tasks.
3) These models have indeed improved substantially over models that use syntax -- getting up to 23% error reduction!
4) But the trends I've just described are mainly when evaluating on text from the same domain as the model was trained on
5) out-of-domain



the improvements have been less impressive.

6) in this setting deep NN models have so far reduced the error rate by only 8%. As I'll describe later there have been some attempts to inject syntax into neural models, but the results on out-of-domain evaluation have been mixed.

7) In this talk I'm going to show that syntax really can help in the midst of deep learning, with the right modeling. Our LISA model obtains an additional 10% reduction in error over the best neural network model when evaluating out-of-domain,

8) and it also improves in-domain SRL, by an additional 6%.

Linguistically-Informed Self-Attention

- **Multi-task learning**
  - Part-of-speech tagging
  - Labeled dependency parsing
  - Predicate detection
  - Semantic role spans & labeling
- **Syntactically-informed self-attention**
  - Multi-head self-attention supervised by syntax

Ok, so now I want to overview the key components of the LISA model:

1) the first component is multi-task learning across 4 related tasks: pos tagging, labeled dependency parsing, predicate detection, and semantic role labeling
2) but we do more than simple parameter sharing: interaction between tasks is through what we call syntactically-informed self-attention
3) where we supervise one attention head to attend to syntactic parents. Through this supervision, the model learns to use that attention head as an oracle providing syntactic information to downstream layers
4) then at test time, we can use either the syntactic structure predicted by LISA, or we can inject syntax from any external parser to improve SRL accuracy without re-training the SRL model
5) we only have to encode the sequence once to predict all these tasks, unlike most previous work.

DRINK

Outline

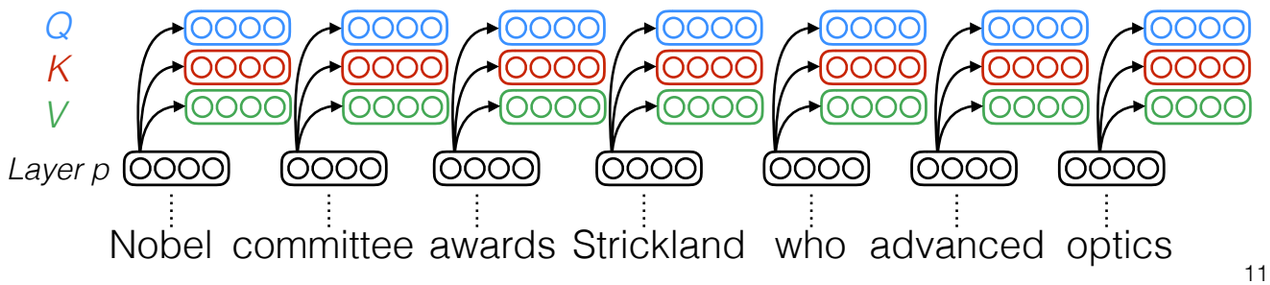
- Want fast, accurate, robust NLU
- PropBank SRL: Who did what to whom?
- 10 years of PropBank SRL
- LISA: Linguistically-informed self attention
  - Multi-head self-attention [Vaswani et al. 2017]
  - Syntactically-informed self-attention
  - Multi-task learning, single-pass inference
- Experimental results & error analysis

Now that I've given a broad overview of what LISA is, I'm going to describe how it works in more technical detail.

1) I'm going to begin by describing multi-head self-attention, since we modify the internals of that mechanism in order to better model information flow between the different tasks
2) For those who are familiar, this is exactly the encoder portion of the Transformer model introduced by Vaswani and others at NIPS last year.

[Vaswani et al. 2017]

Self-attention

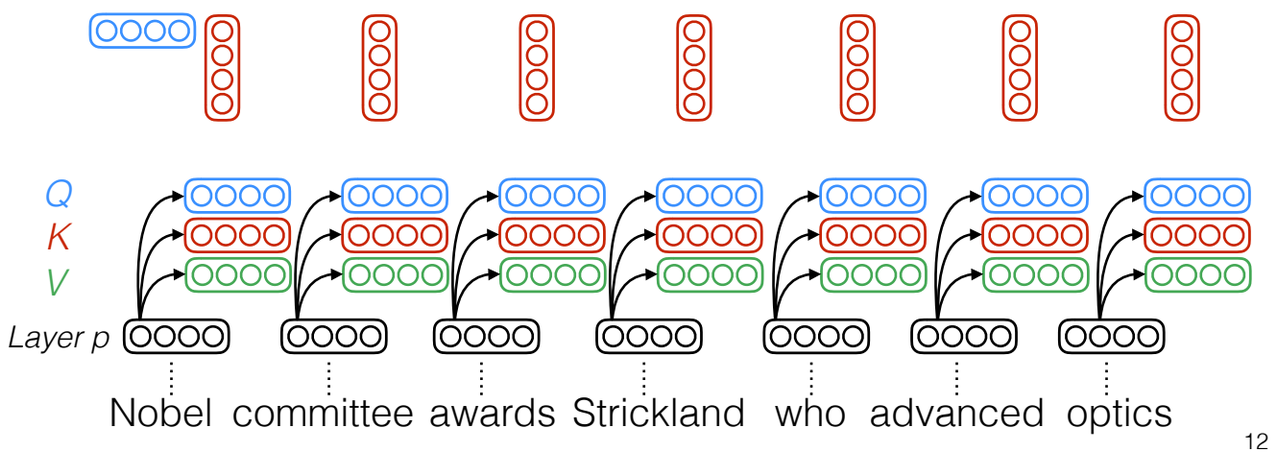


The way that self-attention works: given a sentence and

1) some embedded representations of those tokens, say at layer "p" in the network,
2) the version of self-attention that we're going to use works by first projecting each token into three distinct representations corresponding to each token's role as a "key," "query," or "value" in the self-attention.

Self-attention

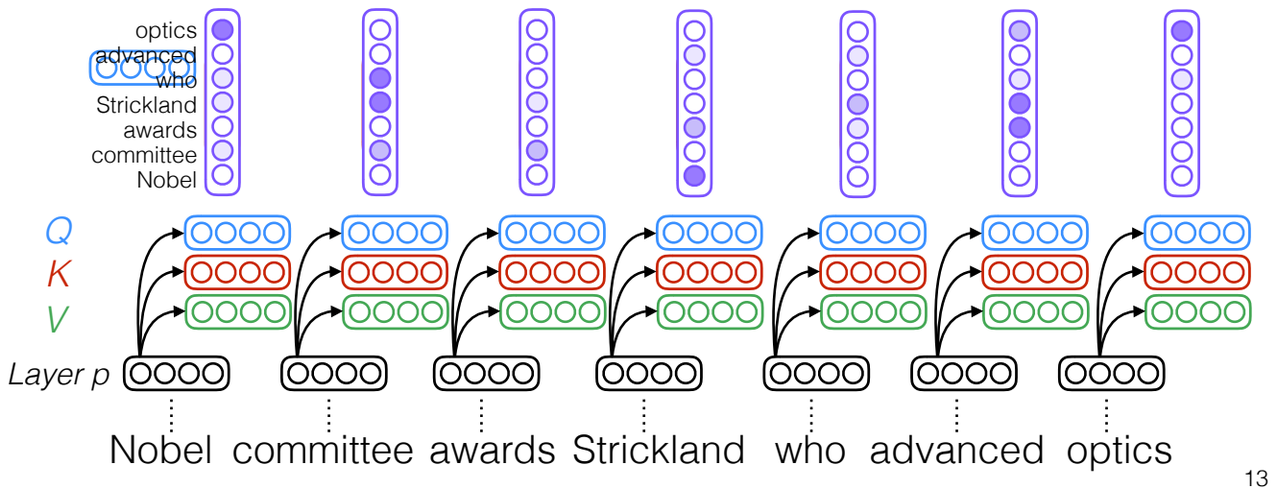
[Vaswani et al. 2017]



For each token, its "query" representation is compared with the "key" representation of every token using a scaled dot product,

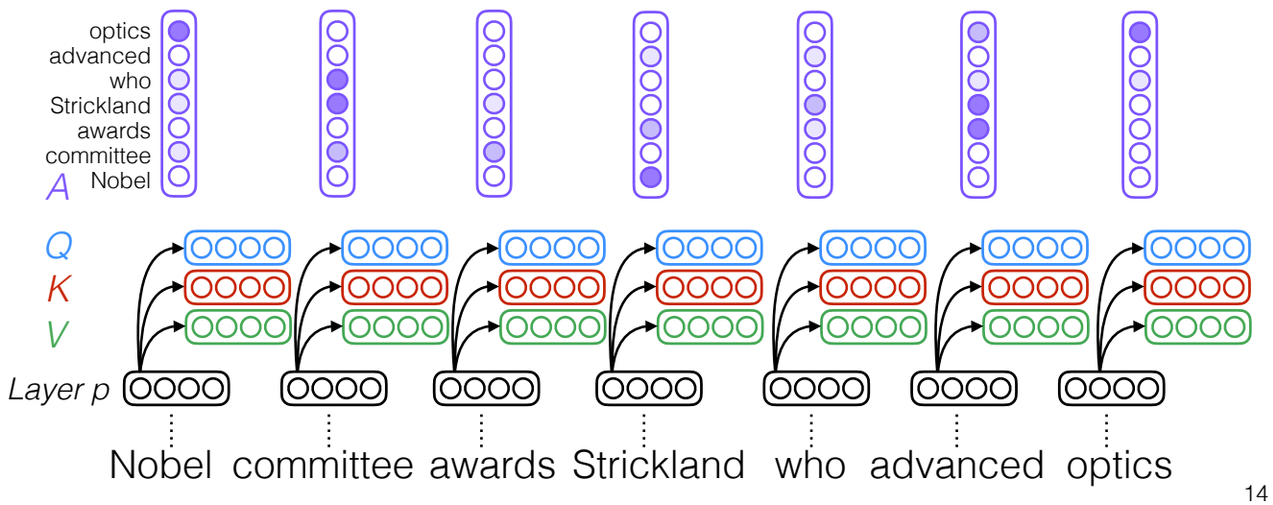
Self-attention

[Vaswani et al. 2017]



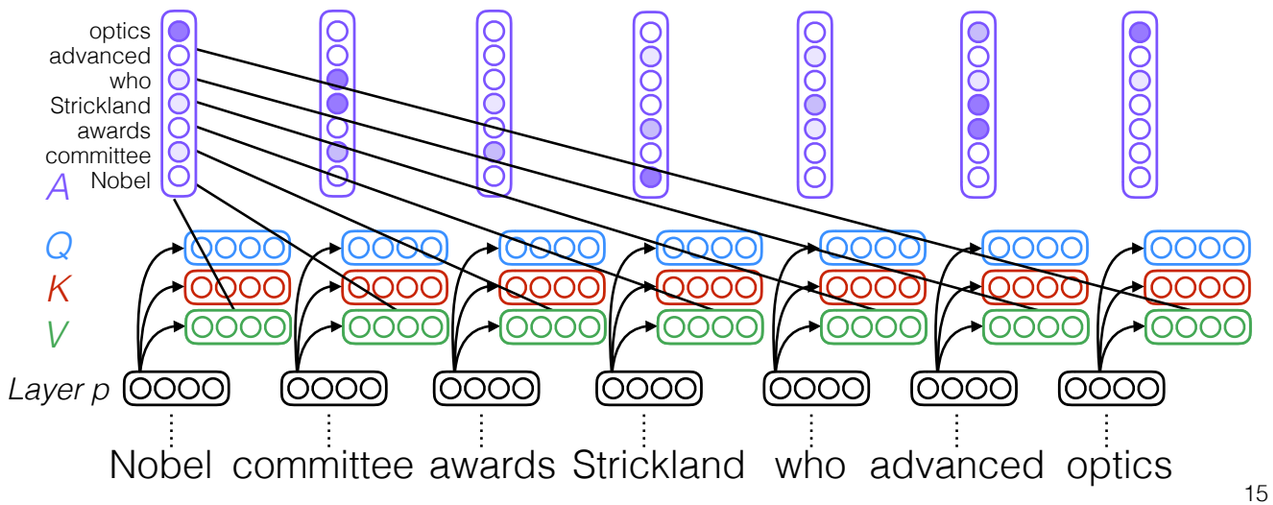
Self-attention

[Vaswani et al. 2017]



[Vaswani et al. 2017]

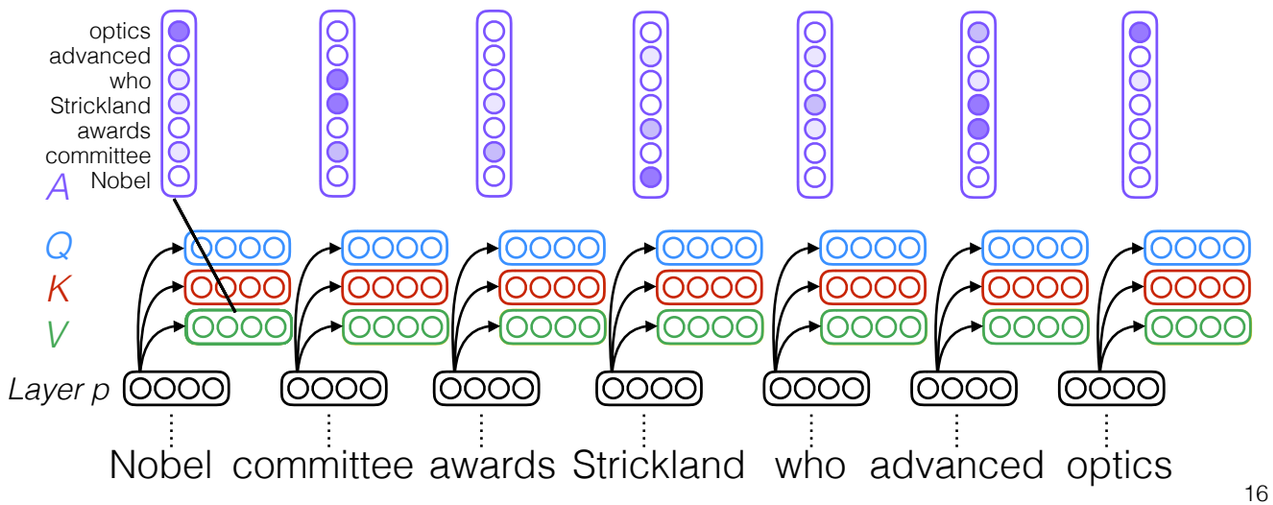
Self-attention



1) which gives for each token a score with respect to every other token in the sentence. These are normalized via the softmax function, giving attention weights between each token with every other token in the sentence.
2) For each token these attention weights are then used to perform a unique weighted average

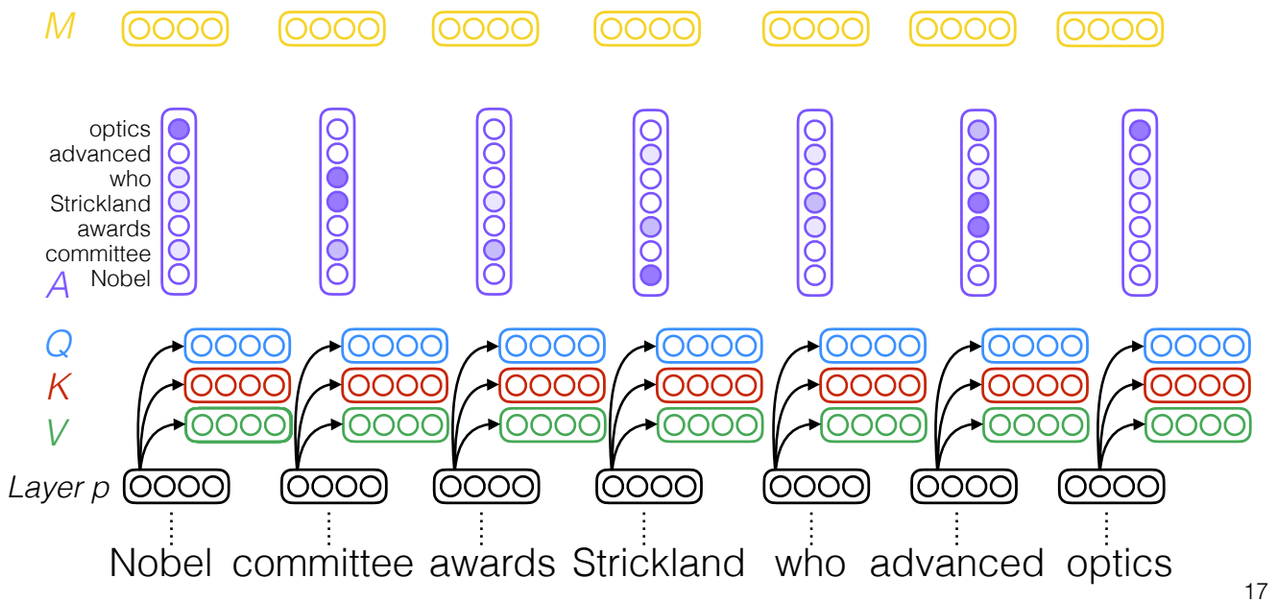
Self-attention

[Vaswani et al. 2017]



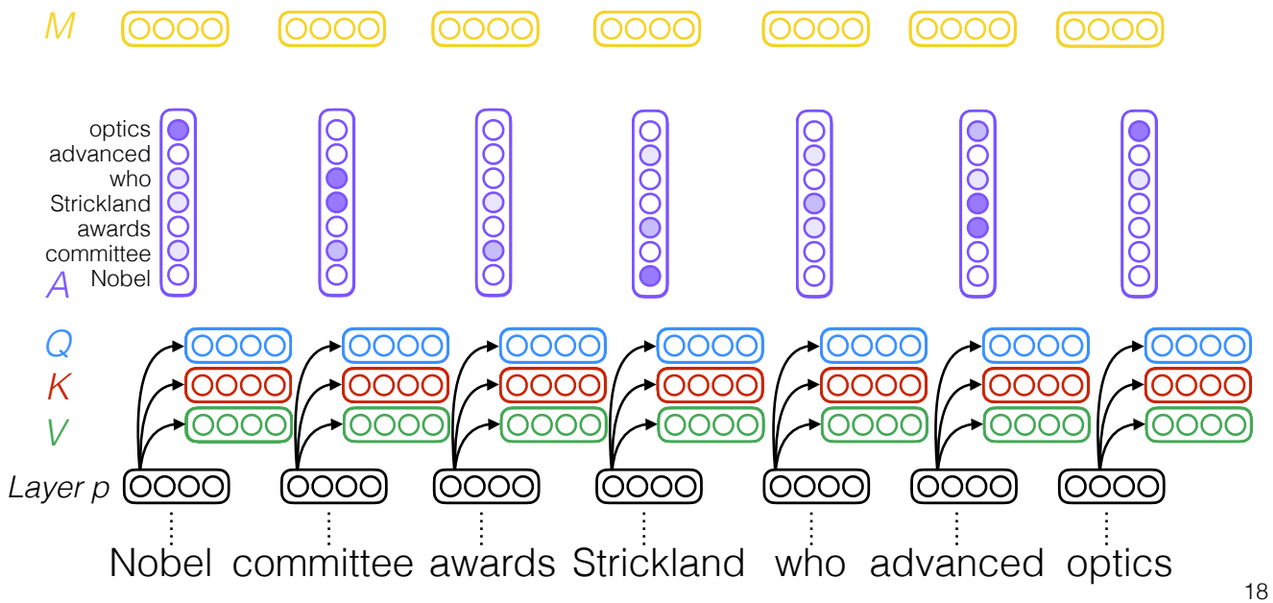
Self-attention

[Vaswani et al. 2017]



[Vaswani et al. 2017]

Self-attention

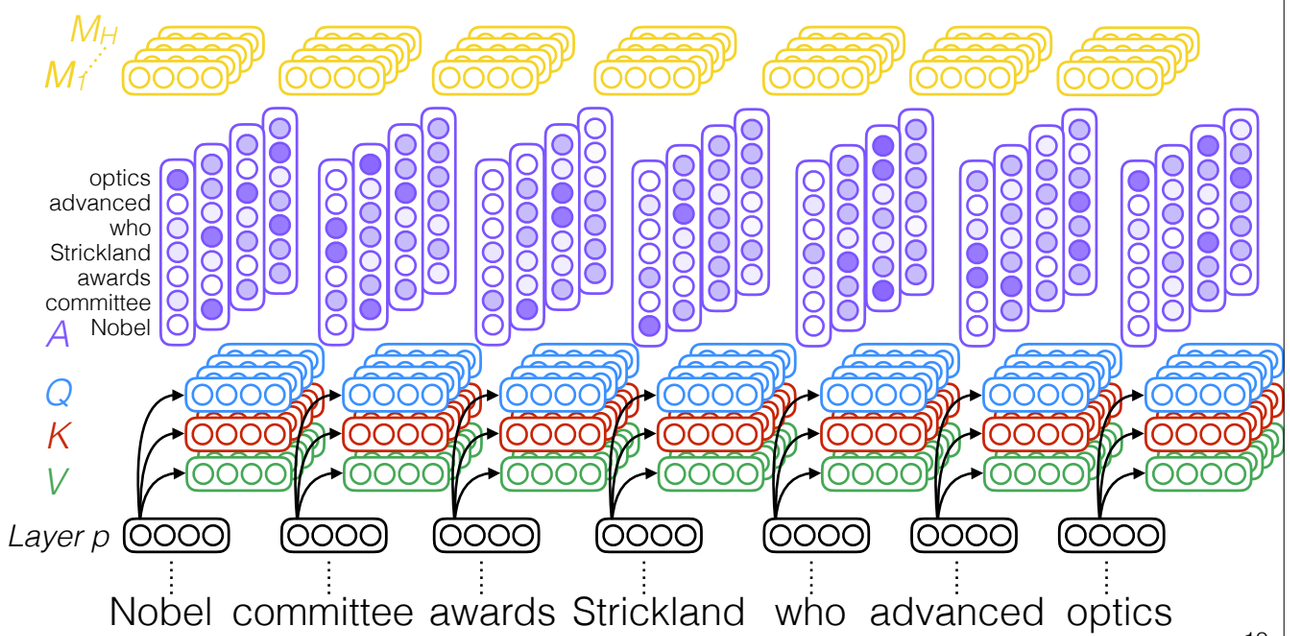


over the "value" representations of all the tokens

Resulting in a new, attended representation for each token. So, this is the basic self-attention model.

[Vaswani et al. 2017]

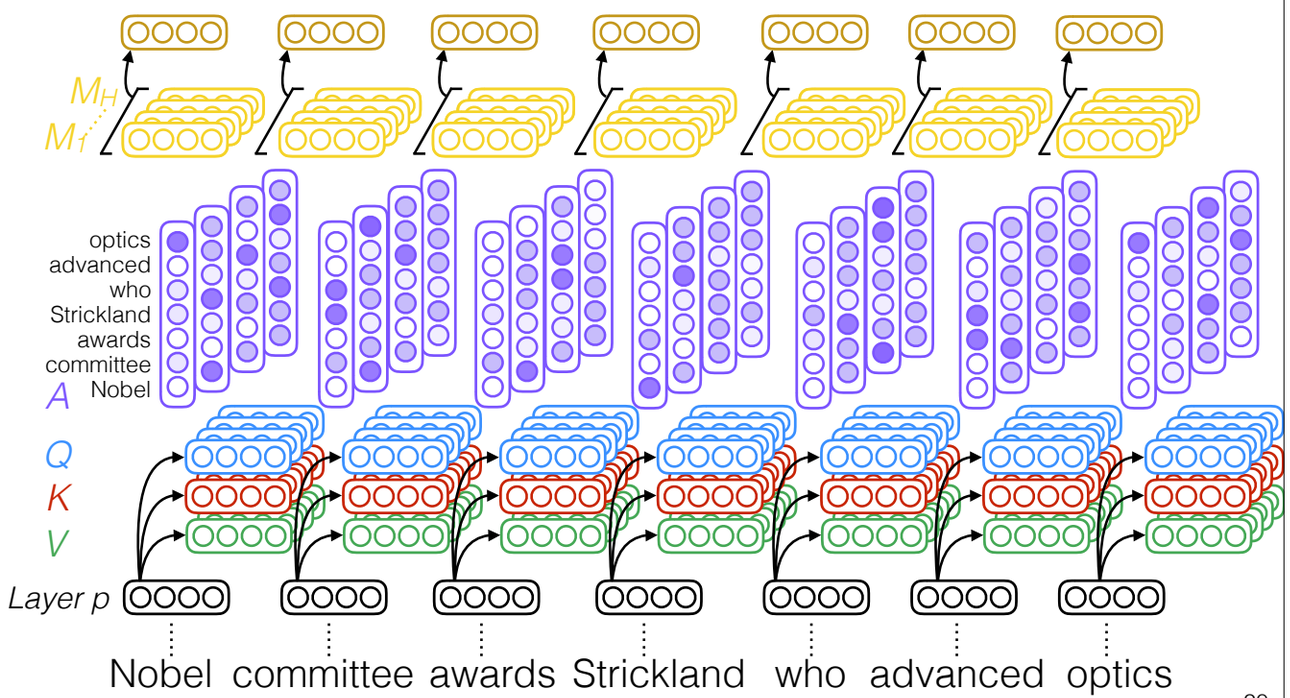
Multi-head self-attention



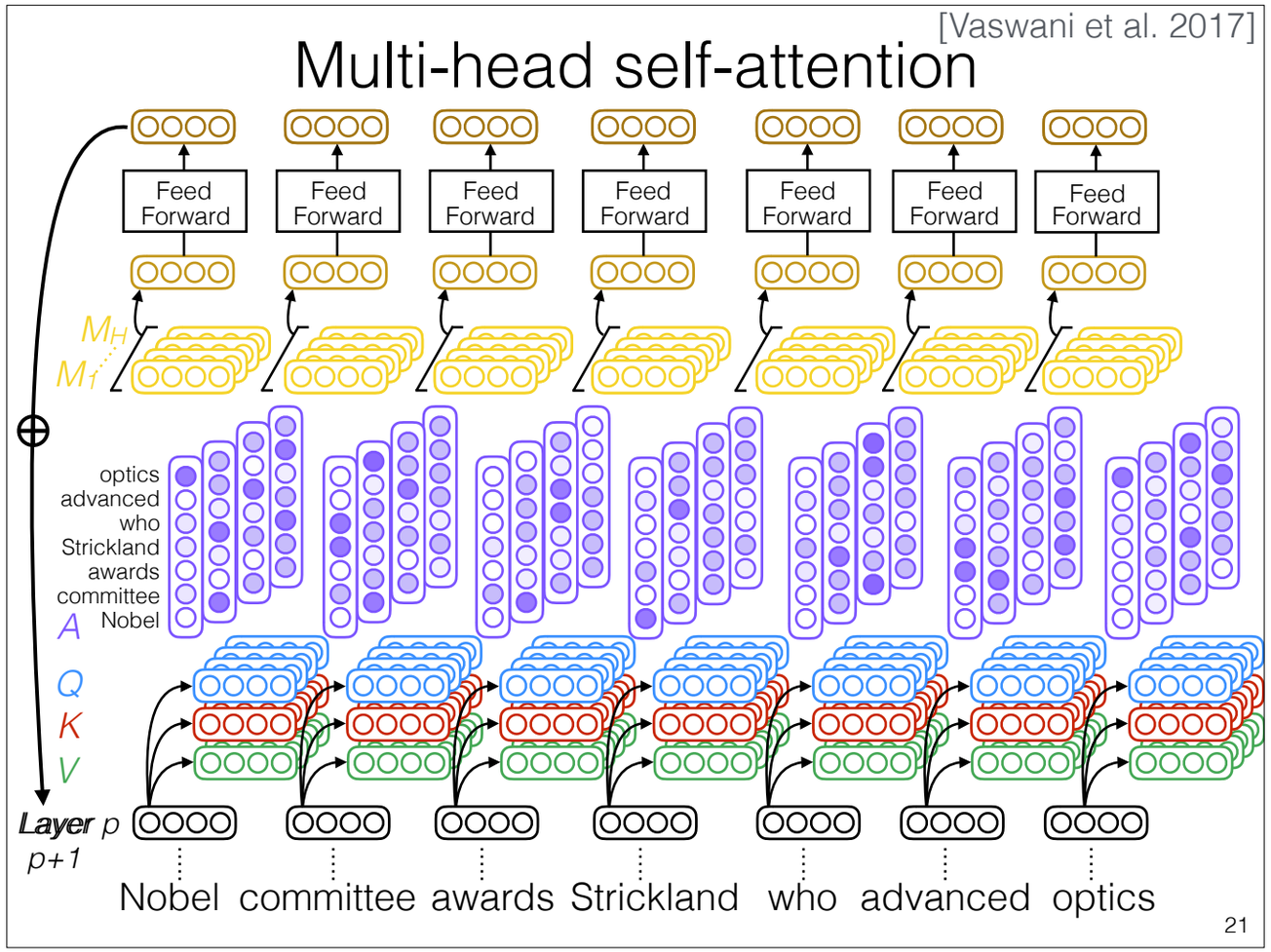
An important aspect of the Transformer model is "multi-head" self-attention, which simply means that at each layer in the network, we actually have H distinct sets of self-attention parameters, normalized separately and learning H distinct self-attention functions.

[Vaswani et al. 2017]

Multi-head self-attention



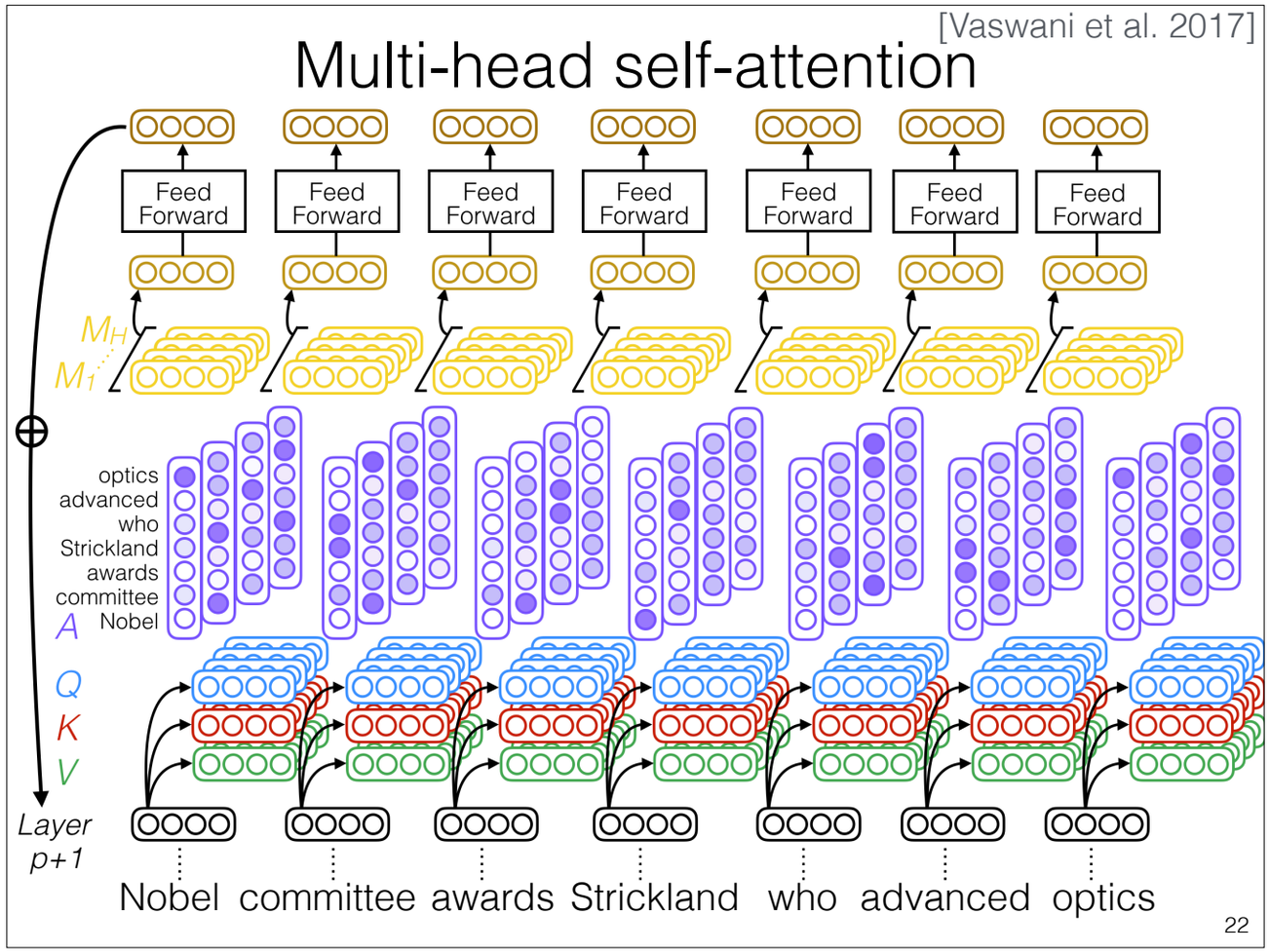
The outputs of each attention head for each token are concatenated.



and that concatenated representation is passed through a feed-forward layer.

1) Finally, this representation is added to the initial input of the layer via a residual connection,

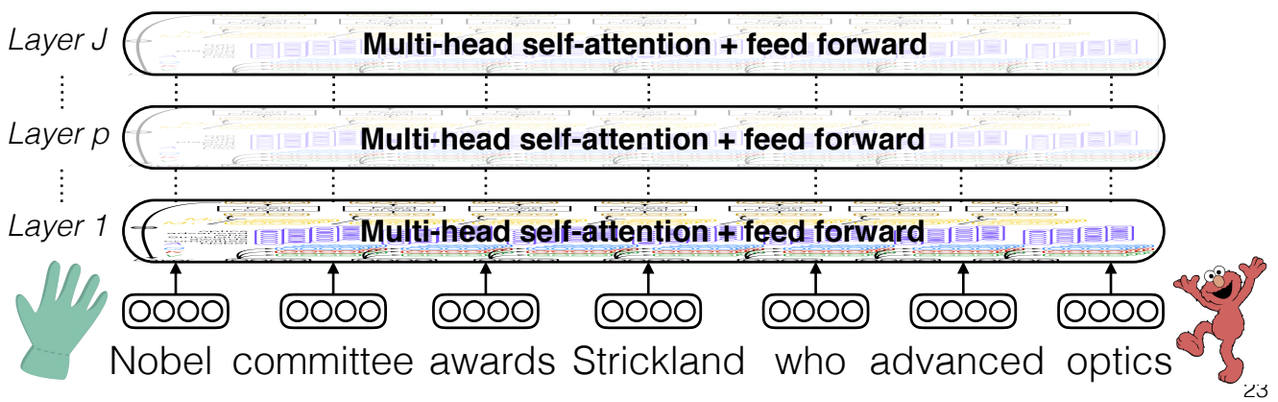
[Vaswani et al. 2017]



Giving the representation at layer p+1, the input to the next layer.
DRINK

[Vaswani et al. 2017]

Multi-head self-attention



We stack J layers of this self-attention,

1) and on the input we use pre-trained word representations. We experiment with both Glove and ELMo embeddings.

Outline

- Want fast, accurate, robust NLU
- PropBank SRL: Who did what to whom?
- 10 years of PropBank SRL
- LISA: Linguistically-informed self attention
  - Multi-head self-attention [Vaswani et al. 2017]
  - Syntactically-informed self-attention
  - Multi-task learning, single-pass inference
- Experimental results & error analysis

Now that I've described how basic multi-head self-attention works, I want to describe how we modify this to incorporate syntactic information.

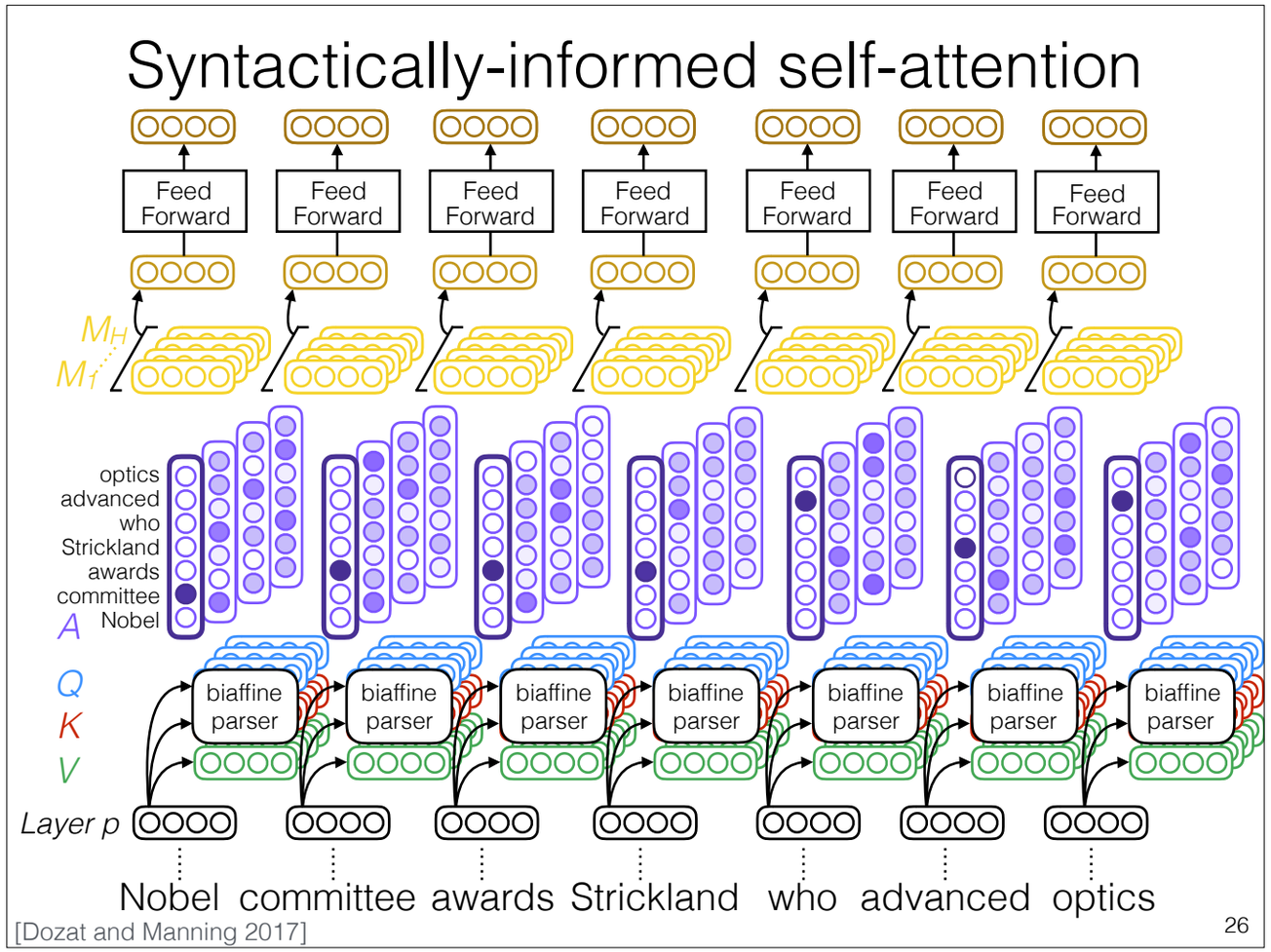
How to incorporate syntax?

- Multi-task learning [Caruana 1993; Collobert et al. 2011]:
  - Overfits to training domain like single-task end-to-end NN.
  - Must re-train SRL model to leverage new (improved) syntax.
- Dependency path embeddings [Roth & Lapata 2016]; Graph CNN over parse [Marcheggiani & Titov 2017]
  - Restricted context: path to predicate or fixed-width window.
- Syntactically-informed self-attention
  - In one head, token attends to its likely syntactic parent(s).
  - Global context: In next layer, tokens observe all other parents.
  - At test time: can use own predicted parse, *OR* supply syntax to improve SRL model without re-training.

First I want to put our approach into a little more context to help explain how we arrived at our technique for integrating syntax into a neural network SRL model.

1) One straightforward approach is to combine SRL and parsing via multi-task learning w/ hard parameter sharing,
2) but just like a single-task end-to-end model this will overfit to the training domain
3) and the model will have to be re-trained from scratch in order to leverage improved syntax models or data
4) More recently others have incorporated syntax into neural models for dependency-based SRL, either through dependency path embeddings or a graph CNN over the parse tree,
5) And while these techniques can leverage new syntax without re-training, they only incorporate limited syntactic context, and it's perhaps for this reason that they have observed mixed results on out-of-domain data
6) Which brings us to our model, syntactically-informed self-attention.
7) This is a really natural way to incorporate syntactic information: each token attends to its likely syntactic parents.
8) This allows the model to learn more global features over the parse, since in subsequent layers, since each token attends to all other tokens, each token observes not only its own parents but those of all the other tokens.
9) And it combines the best of both worlds: as in typical multi-task learning, at test time the model can use its own predicted parse, OR, like methods that use dependency paths or graph CNNs, an externally-generated parse can be supplied to improve SRL performance without re-training.

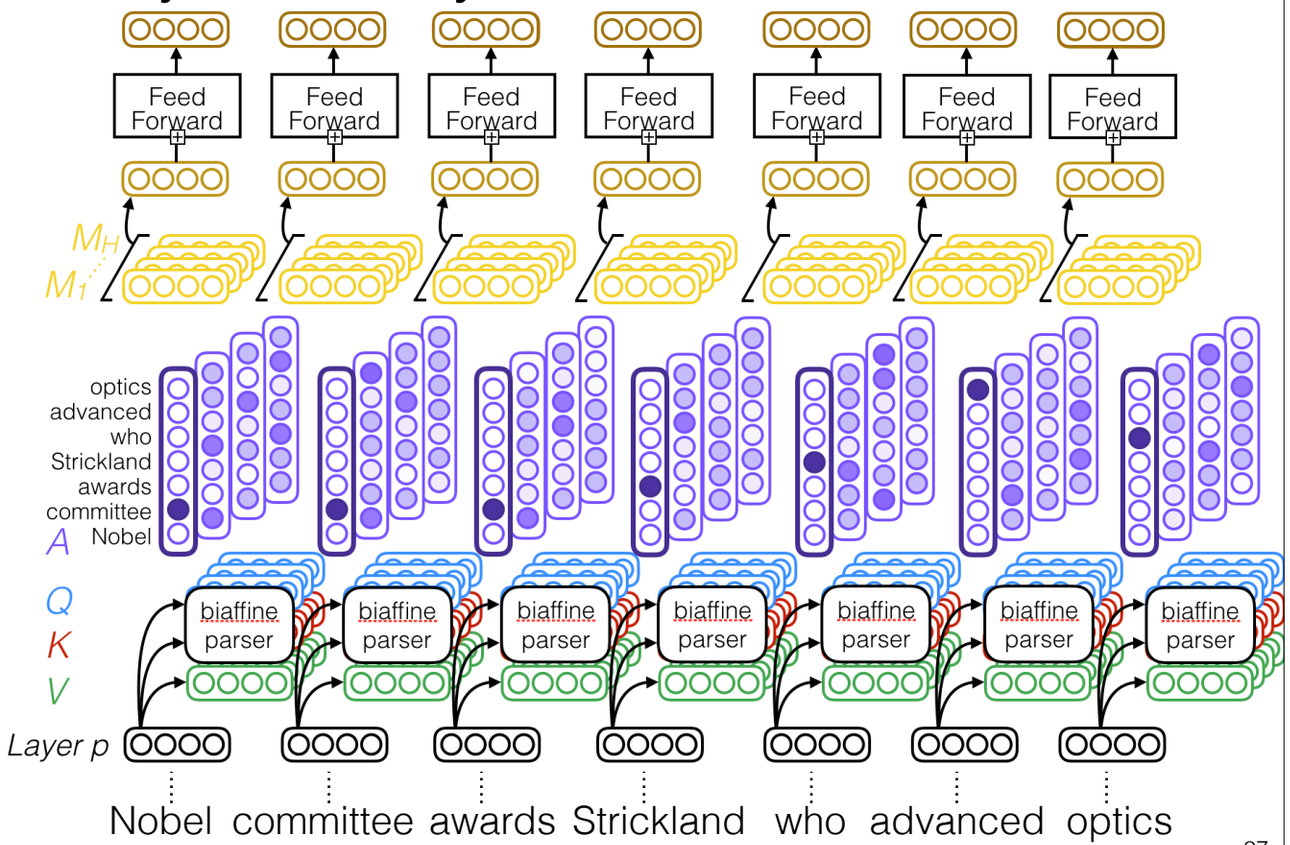
DRINK



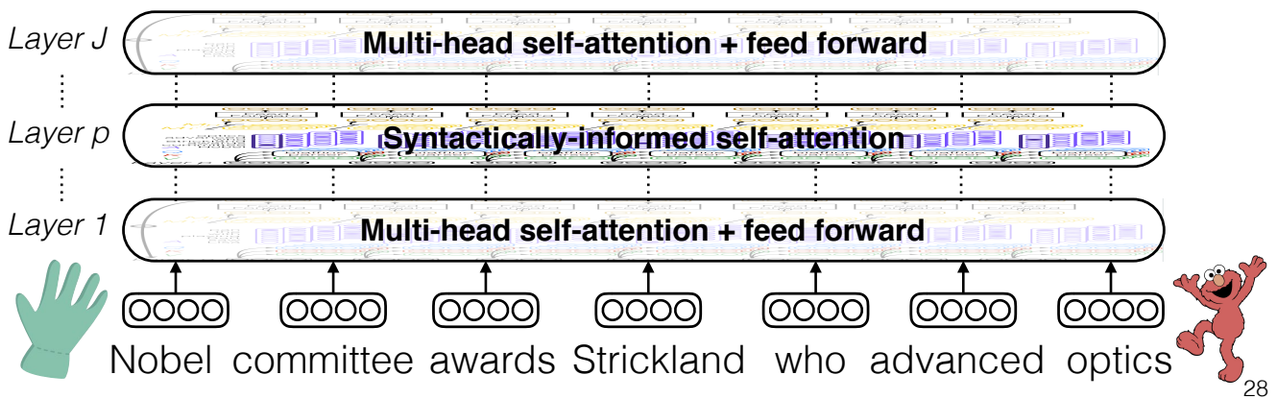
Now I'm going to depict exactly how this works. Returning to our depiction of multi-head self-attention,

1) Typically, each attention head is left to learn its own attention function.
2) Instead, we train one attention head to attend to syntactic parents
3) We do this by replacing the "query" and "key" mechanism in that attention head by the biaffine parsing mechanism of Dozat and Manning, which produces a matrix of edge scores between each pair of tokens. This matrix is simply dropped in to replace the attention matrix generated by query-key pairs
4) That's all there is to it. Otherwise (CLICK)
5) DRINK

Syntactically-informed self-attention



Syntactically-informed self-attention



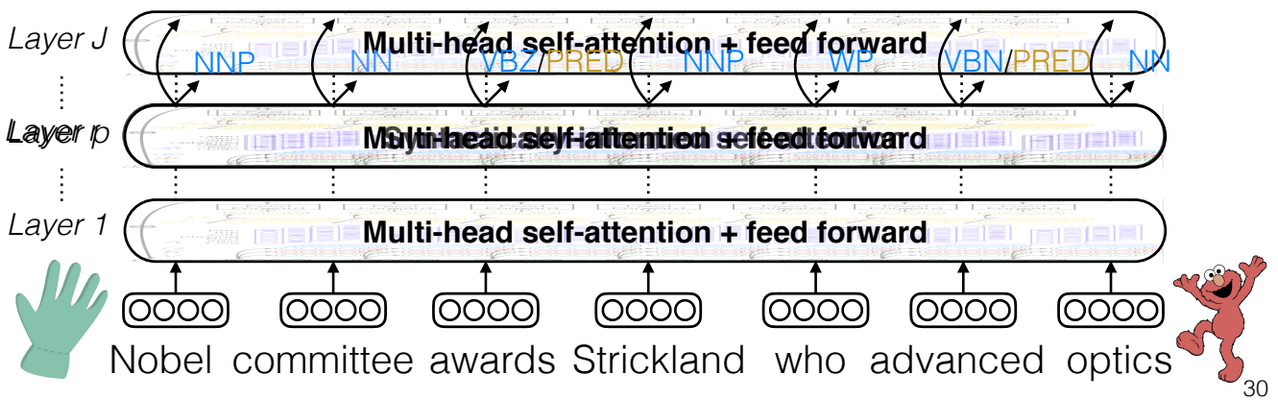
1) this layer acts just like any other self-attention layer. We choose one layer in the network as the syntactically-informed layer.

Outline

- Want fast, accurate, robust NLU
- PropBank SRL: Who did what to whom?
- 10 years of PropBank SRL
- LISA: Linguistically-informed self attention
  - Multi-head self-attention
  - Syntactically-informed self-attention
  - Multi-task learning, single-pass inference
- Experimental results & error analysis

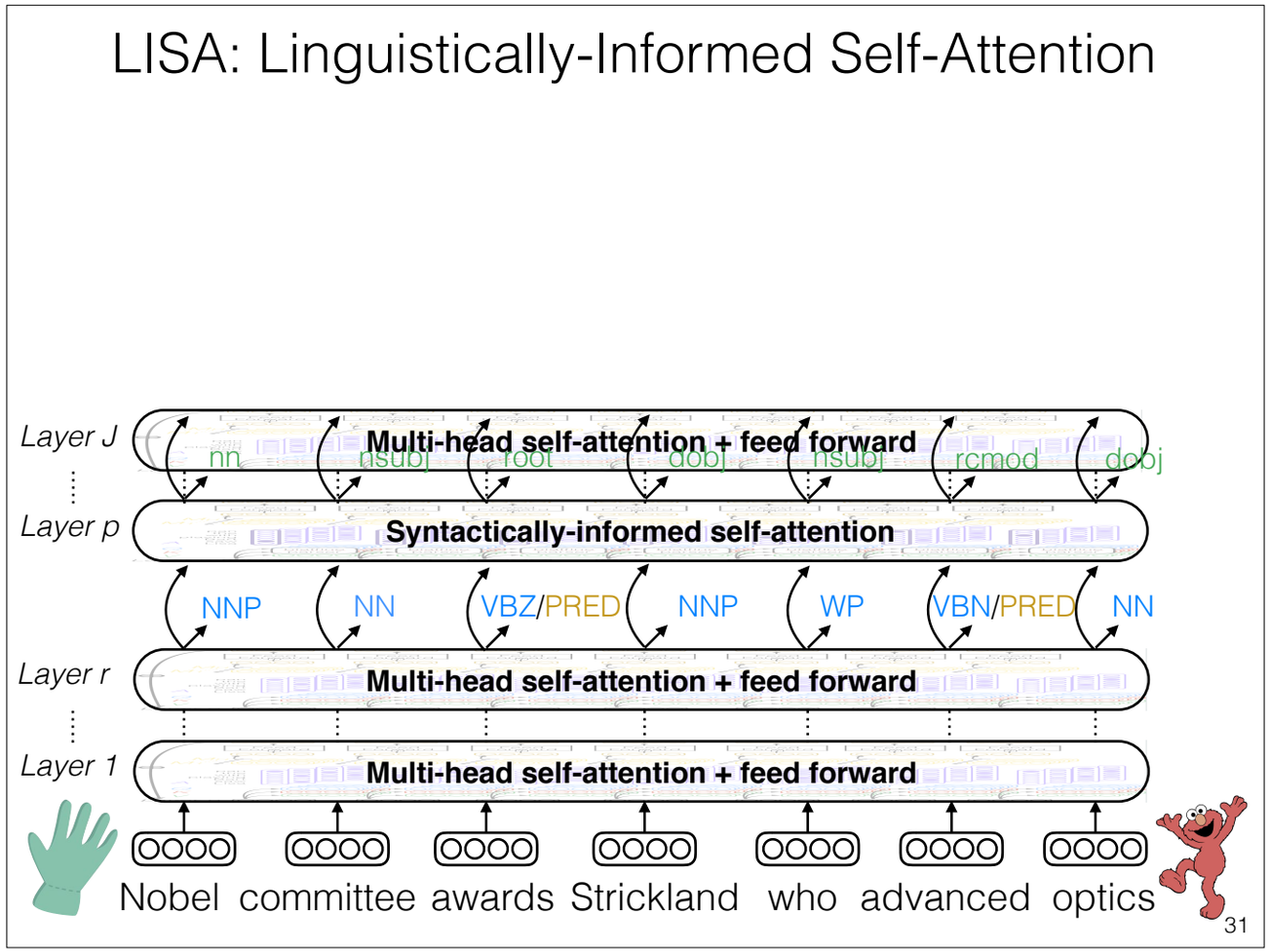
Now that I've described how we incorporate syntax into the attention mechanism,
1) I want to describe the other important aspect of LISA, which is how we do all these tasks, including predicate detection, in just a single pass through the network.

LISA: Linguistically-Informed Self-Attention



Here's the network I've described so far.

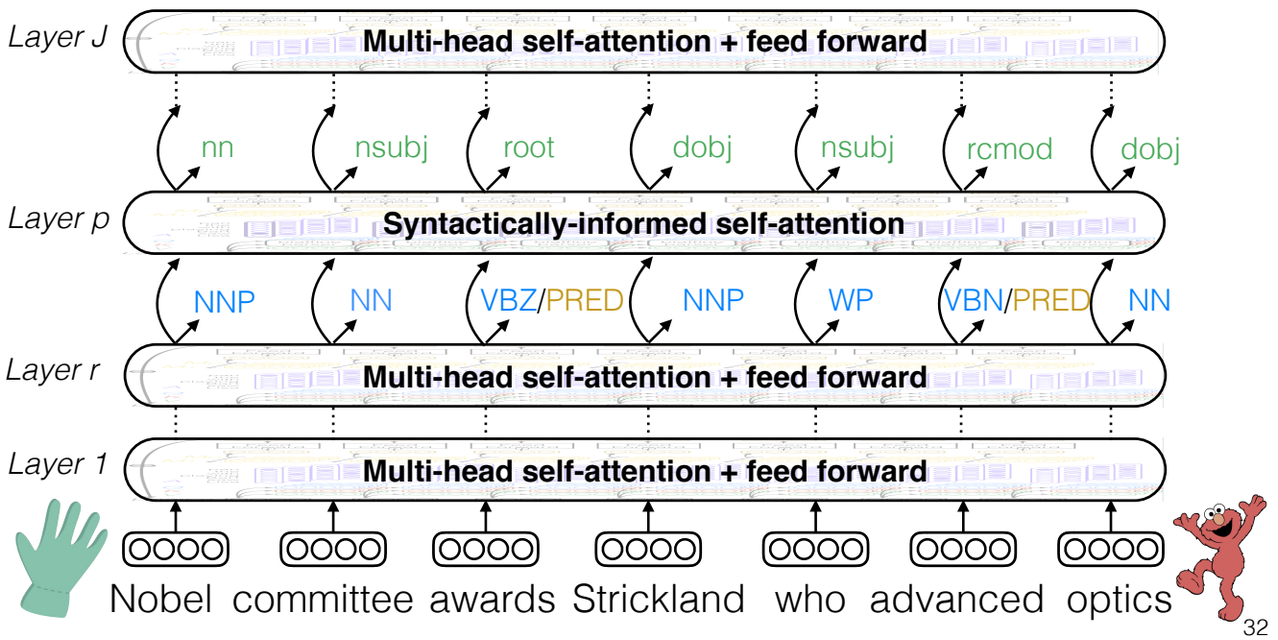
1) At an early layer, exactly which layer is left as a hyperparameter,
2) we predict parts-of-speech and predicates.



Since the two labels are highly related, and in order to reduce complexity of training the model, we predict into the joint, cross-product space of part-of-speech and binary predicate labels.

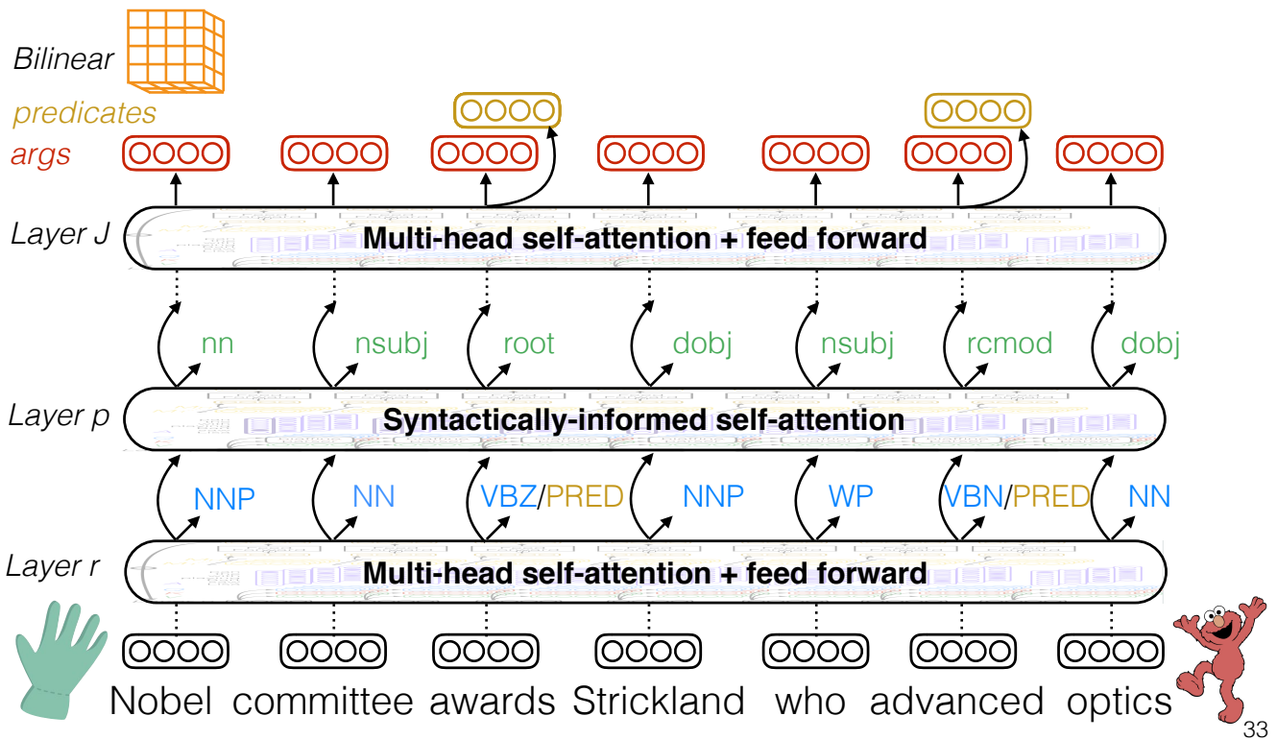
1) At the syntactically-informed layer, following Dozat and Manning

LISA: Linguistically-Informed Self-Attention



We share the parameters of the biaffine parser to also predict dependency labels.
(DRINK)
Now I'm going to demonstrate how we do semantic role labeling with respect to each predicate. First I'm going to remove this bottom layer to make some room on the slide.

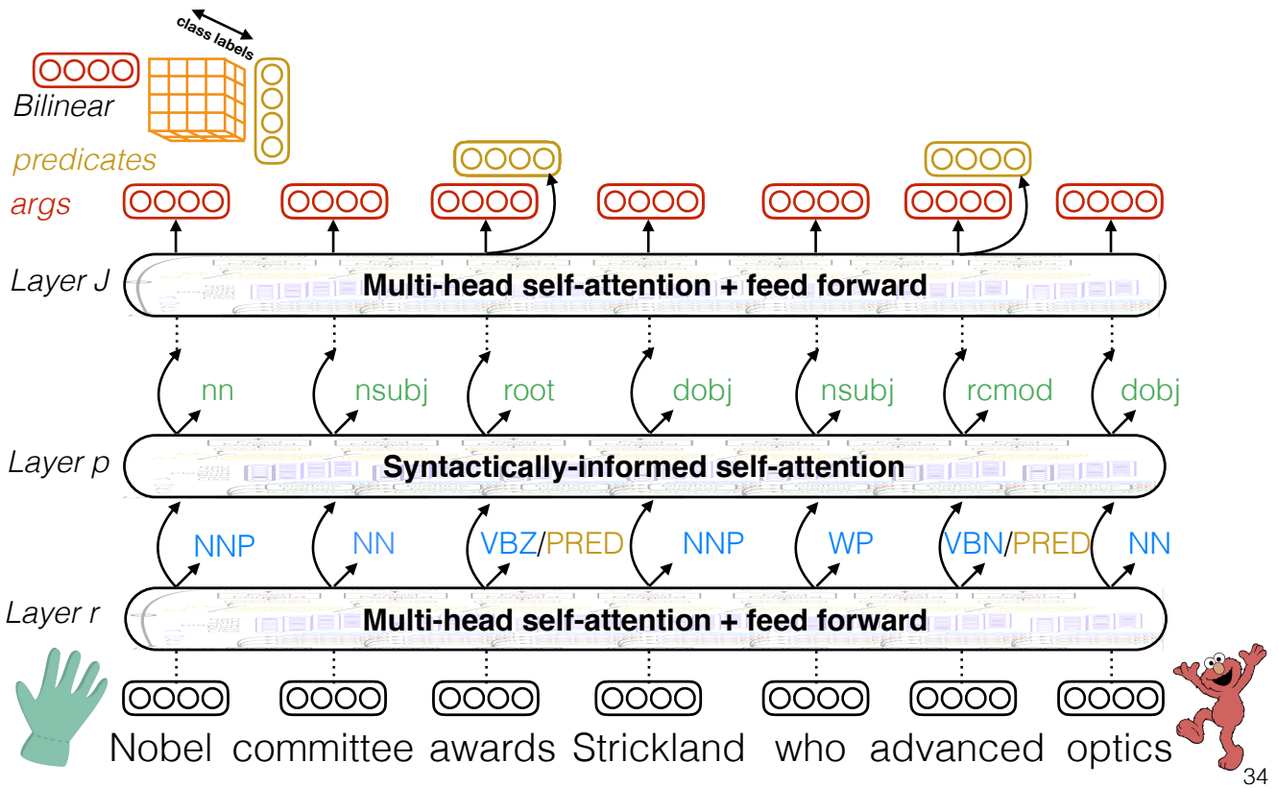
LISA: Linguistically-Informed Self-Attention



First, we project each final-layer token representation into an "argument"-specific representation of that token,

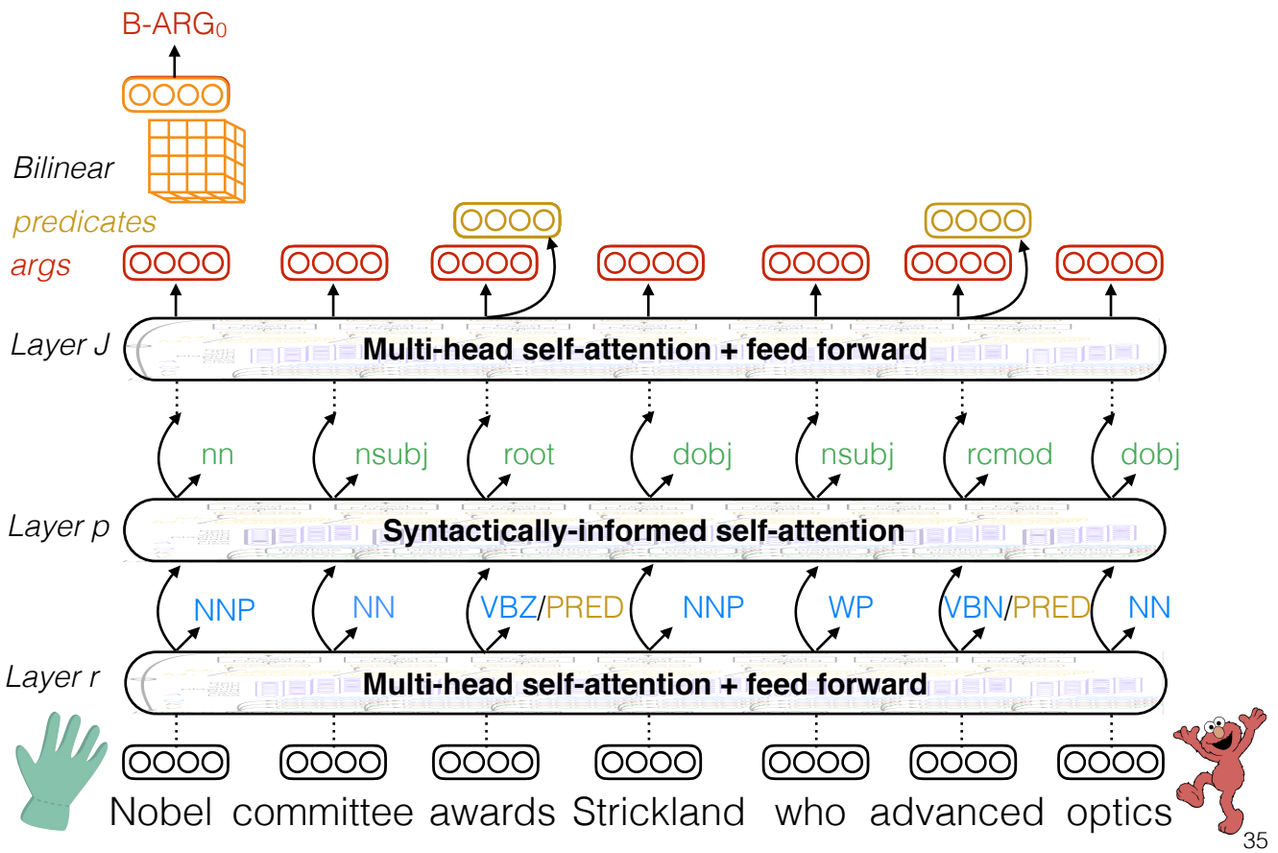
1) and for each token that was predicted to be a predicate, we project that token to a "predicate"-specific representation.
2) We then use a bilinear operator

LISA: Linguistically-Informed Self-Attention



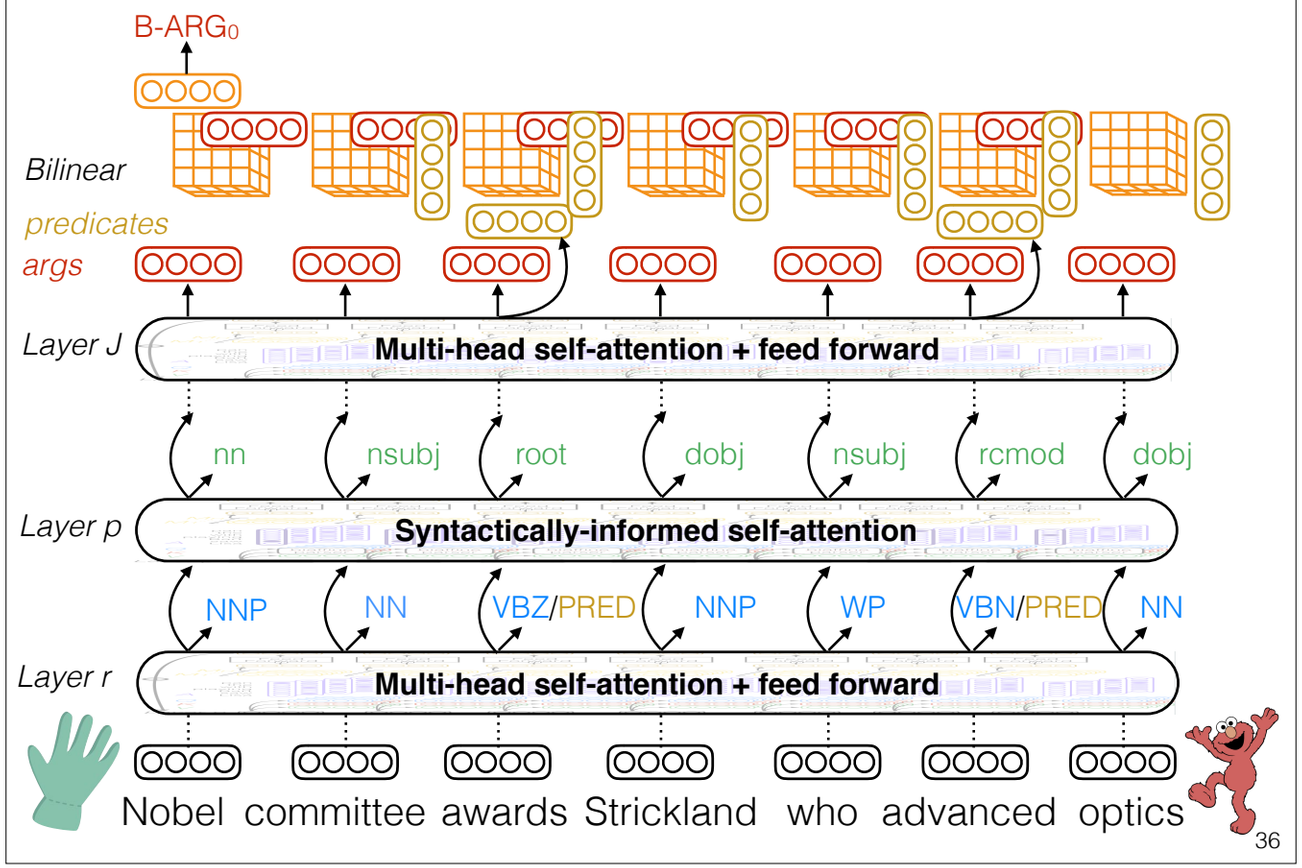
1) to score the compatibility between each token with each of the predicates. Here I'm scoring "Nobel" with the predicate "awards."
2) We actually want to score, for each token, what its LABEL is with respect to that predicate, so we have a bilinear matrix for each SRL class label,

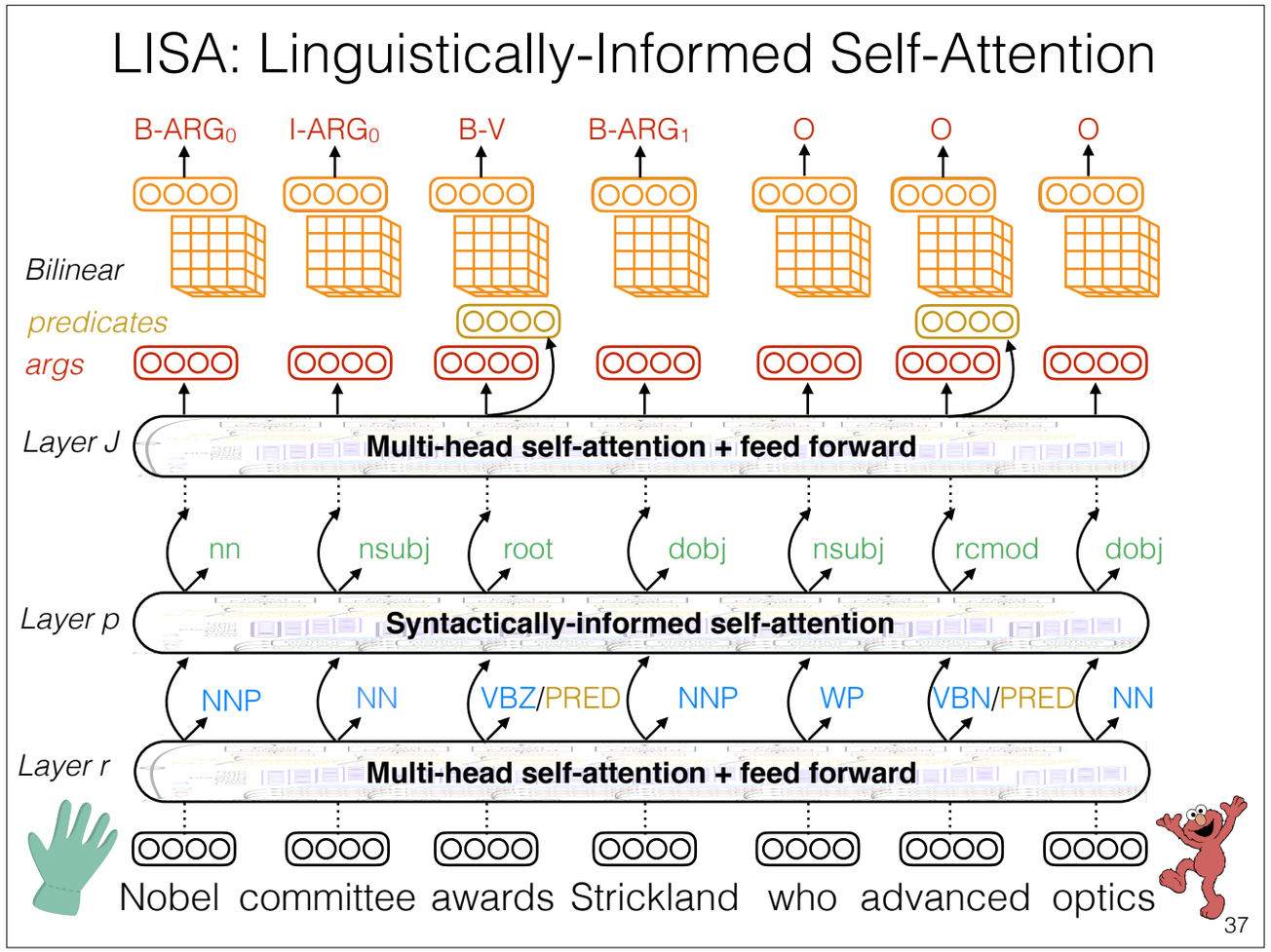
LISA: Linguistically-Informed Self-Attention



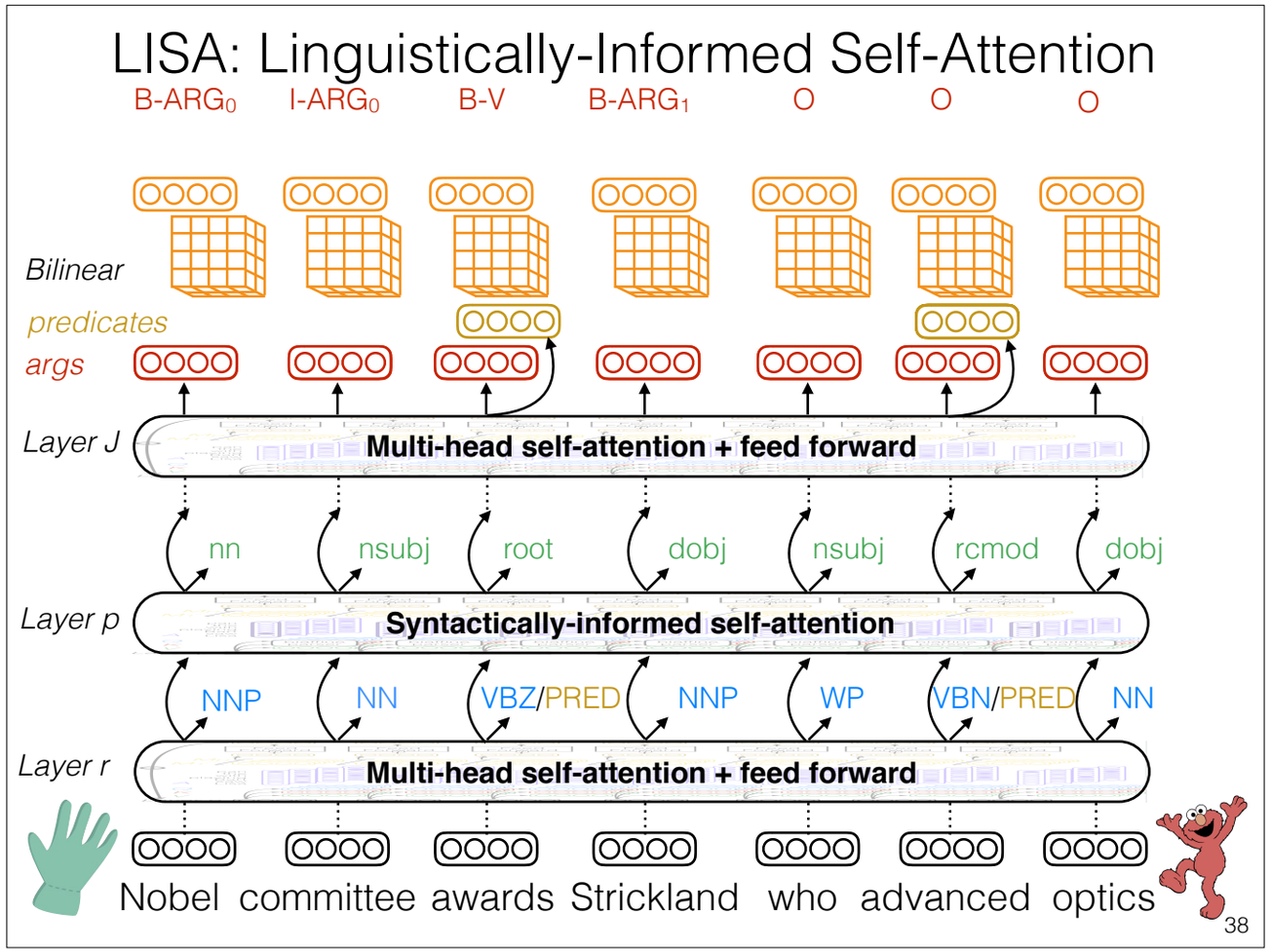
and this gives us SRL label predictions w/ respect to that predicate

LISA: Linguistically-Informed Self-Attention



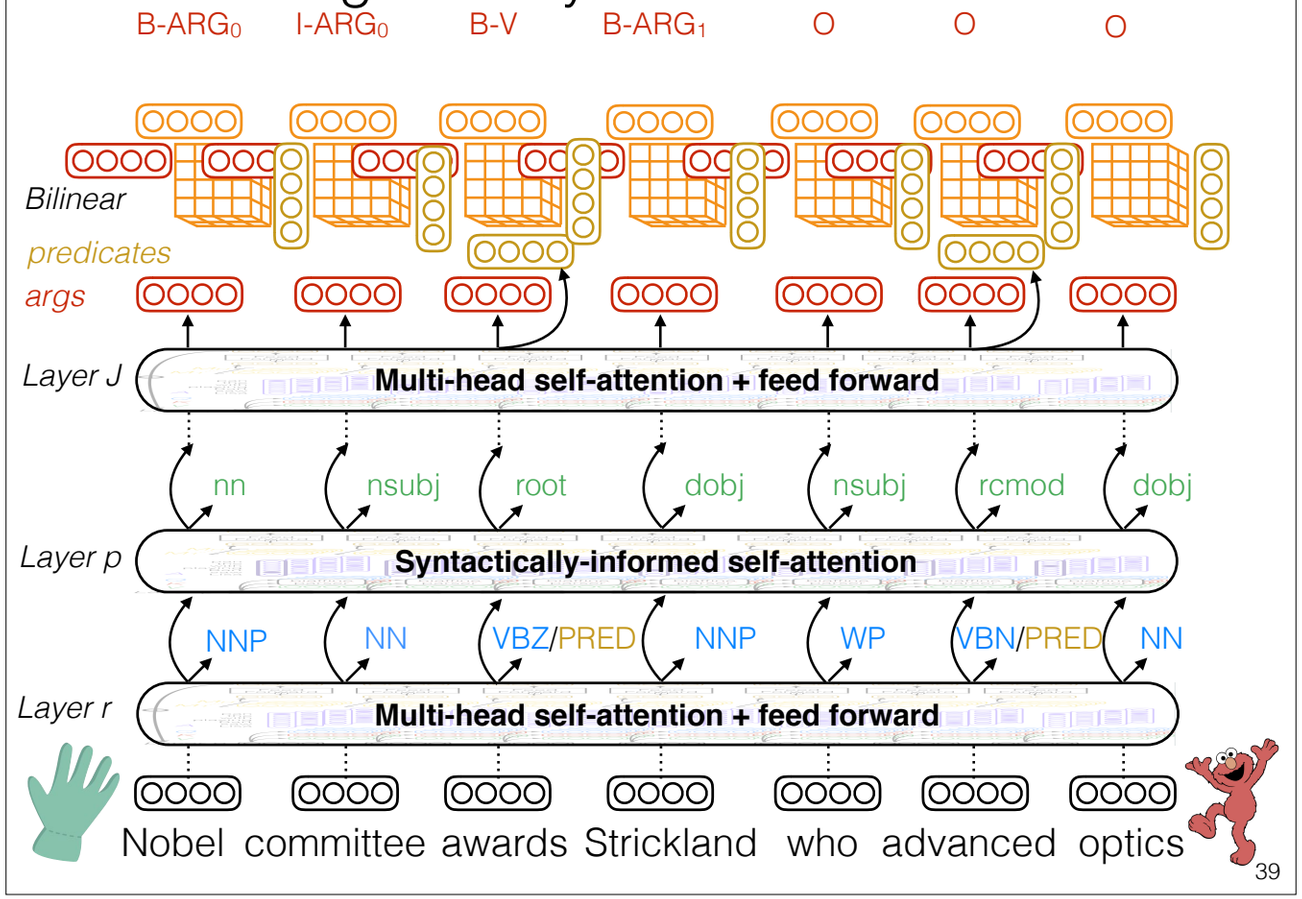


In this same way we score each token with the first predicate to get a labeling of the sequence with respect to that predicate.

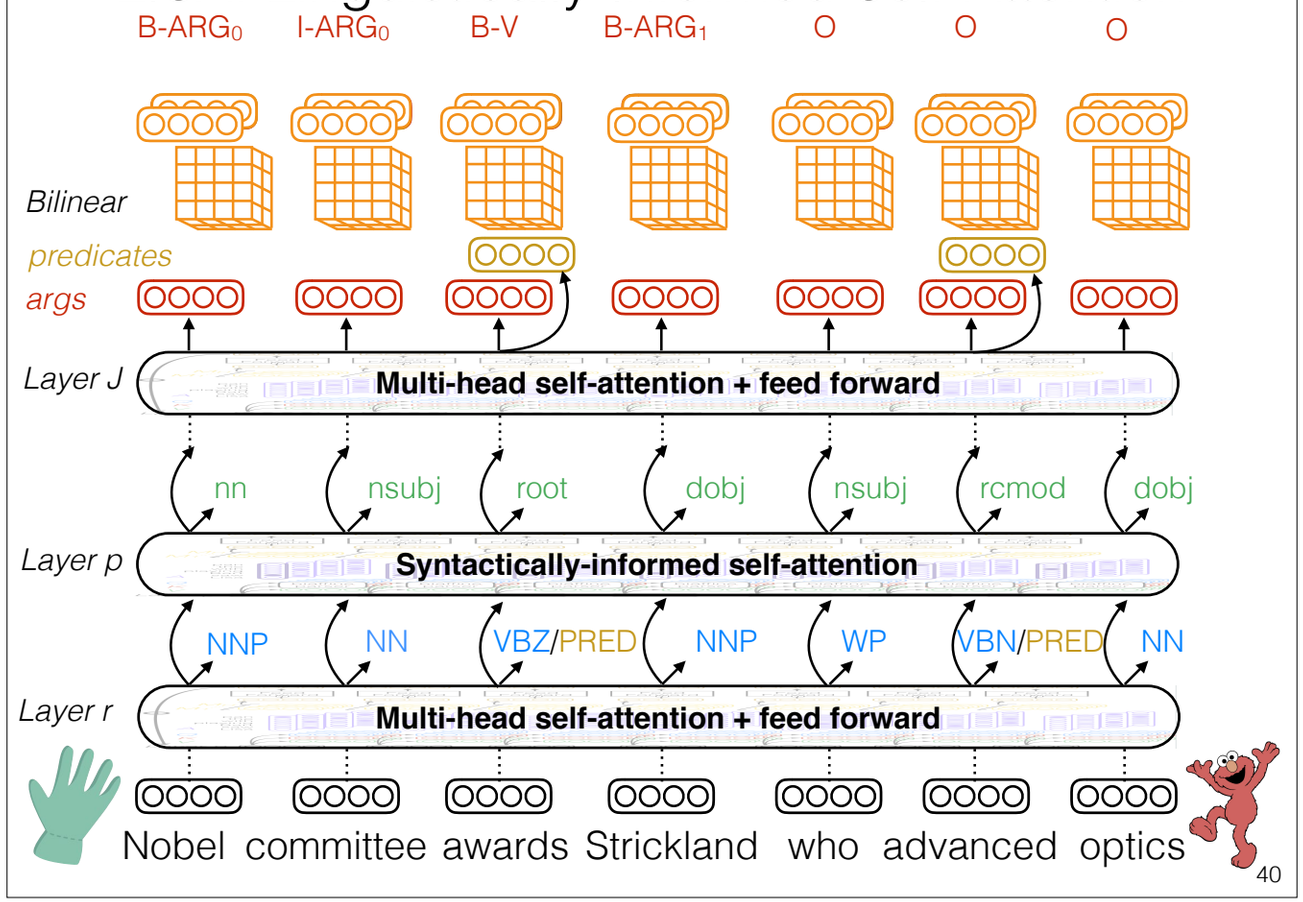


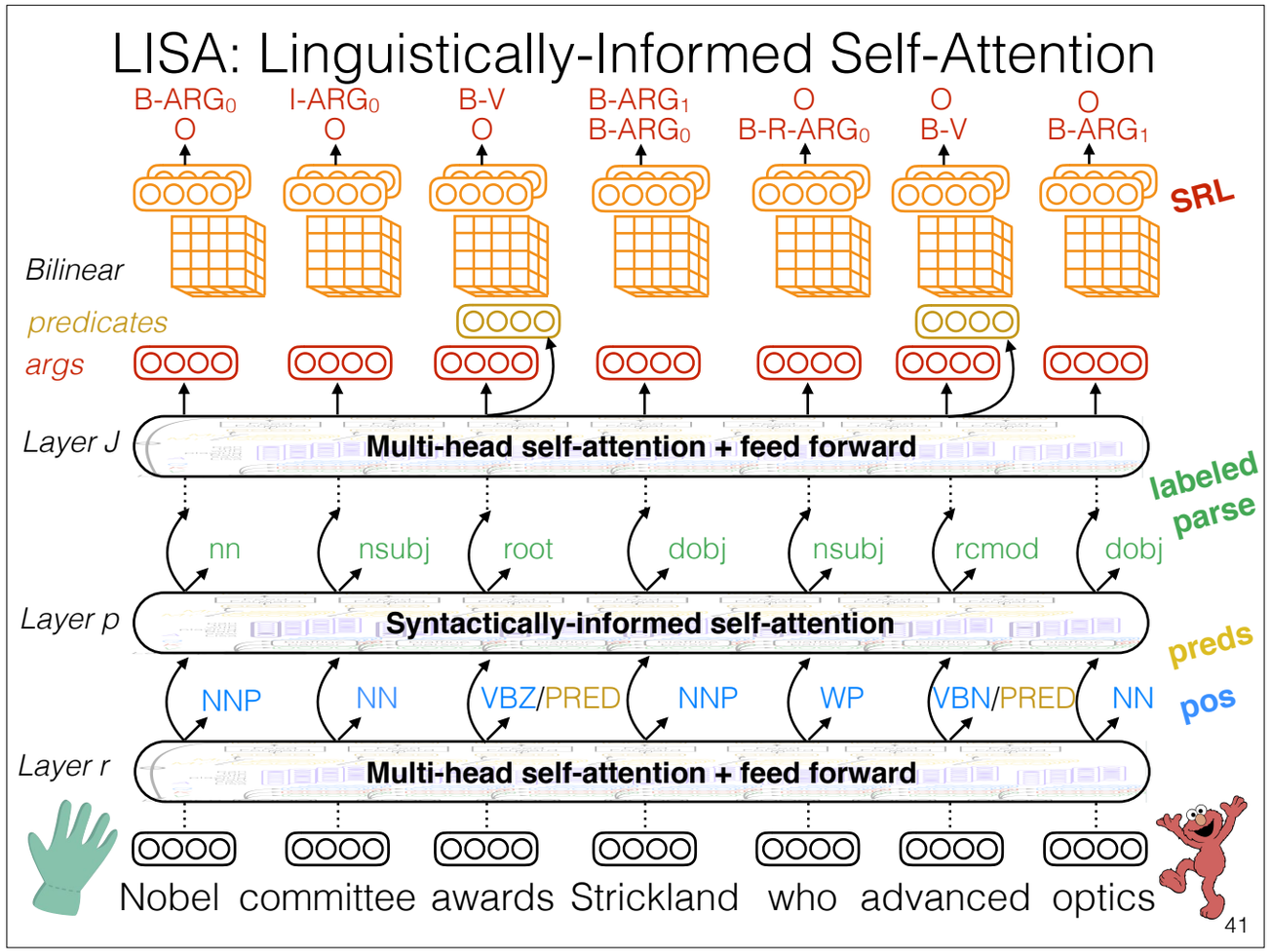
And we do the same for each predicted predicate, in this case we have one other predicate, "advanced" (CLICK earlier)

LISA: Linguistically-Informed Self-Attention



LISA: Linguistically-Informed Self-Attention





giving us a semantic role labelling of the sentence with respect to each predicted predicate. And in practice this is really fast, because we can score all predicates with all tokens for an entire batch using just two big matrix multiplies.

And this is the LISA model in its entirety --

1) multitask learning across four related tasks,
2) with the signal for syntax shared in this special way through the attention mechanism.

DRINK

Outline

- Want fast, accurate, robust NLU
- PropBank SRL: Who did what to whom?
- 10 years of PropBank SRL
- LISA: Linguistically-informed self attention
  - Multi-head self-attention
  - Syntactically-informed self-attention
  - Multi-task learning, single-pass inference
- Experimental results & error analysis

Now that I've described the model I want to highlight some of our experimental results.

Experimental results

	CoNLL-2005	CoNLL-2012
domains	Train, dev: news Test: news, novels	Train, dev, test: 7 domains (news, telephone, bible, ...)
word embeddings	GloVe [Pennington et al. 2014] ELMo [Peters et al. 2018]	GloVe [Pennington et al. 2014] ELMo [Peters et al. 2018]
predicates	predicted; gold	predicted
baselines	He et al. 2017 He et al. 2018 Tan et al. 2018	He et al. 2018
our models	SA LISA LISA+D&M, +Gold	SA LISA LISA+D&M, +Gold

We evaluate our model primarily on two benchmark datasets for semantic role labeling: the CoNLL-2005 and 2012 shared tasks.

1) In this presentation, for the sake of time, I'm going to focus on only a subset of the CoNLL-2005 experiments, but I encourage you to check out the paper if you're interested in the full results and analysis
2) We present in-domain experiments on news data and out-of-domain experiments on novels from the Brown corpus
3) We experiment with both 100 dimensional GloVe embeddings and ELMo contextualized word representations
4) Today I'm going to present only experiments in the more challenging setting using predicted predicates, though in the paper we also run experiments using gold predicates to compare to more prior work
5) Our baselines are the state-of-the-art models on PropBank SRL, none of which use syntax
6) And we look at three versions of our own model:
7) the SA model, which is the LISA model except with no syntax;
8) the LISA model when using its own predicted parses
9) and the LISA model with parses from an external Dozat and Manning parser, as well as with gold parses, in order to see what the upper bound is for how much SRL can benefit from parsing in our model.

re: baselines: "these are the current"

Experimental results: CoNLL-2005


	GloVe		ELMo	
	in-domain	**out-of-domain**	**in-domain**	**out-of-domain**
He et al. 2017	82.7	70.1	---	---
He et al. 2018	82.5	70.8	86.0	76.1
SA	83.72	71.51	86.09	76.35
LISA	83.61	71.91	86.55	78.05
+D&M	84.99	74.66	86.90	78.25
	+2.49 F1	+3.86 F1	+0.9 F1	+2.15 F1

94.9 UAS 90.3 UAS 96.6 UAS 93.4 UAS

?

These are the previous state-of-the-art numbers on the CoNLL-2005 shared task.

1) In-domain our SA model already out-performs the state-of-the-art when using Glove embeddings, and performs comparably to the state-of-the-art with ELMo representations
2) We see a similar trend when evaluating out-of-domain
3) In-domain our LISA model w/ LISA parses performs comparably to the syntax-free SA model. We think this suggests that even without syntactic supervision the SA model is able to learn information equivalent to a reasonably accurate parser --
4) with glove embeddings the LISA model achieves an unlabeled attachment score around 95,
5) and with ELMo above 96.
6) Out-of-domain, we see that with glove embeddings, the LISA parser performs comparably to the SA model, while the LISA model using ELMo performs much better than the syntax-free model.
7) This difference is due to the higher accuracy of the LISA parser when using ELMo embeddings: about 3 points higher out-of-domain. At this high accuracy, the parse *is* providing information that the model can use to generalize better in the out-of-domain evaluation, so we see here that having this explicit representation of syntax does help over even a very strong end-to-end NN model.
8) Feeding in the external D&M parse at test time gives the best results, particularly w/ Glove embeddings: increase over the previous state-of-the-art by nearly 2.5 F1 in-domain and
9) more than 3.5 F1 out-of-domain
10) And the fact that the in-domain improvement using elmo is so small compared to out-of-domain is actually an interesting result, it suggests that ELMo representations are doing a pretty good job of learning the useful parts of syntax for this task, though they seem to be overfitting to the data domain that the model is trained on.

Experimental results: CoNLL-2005

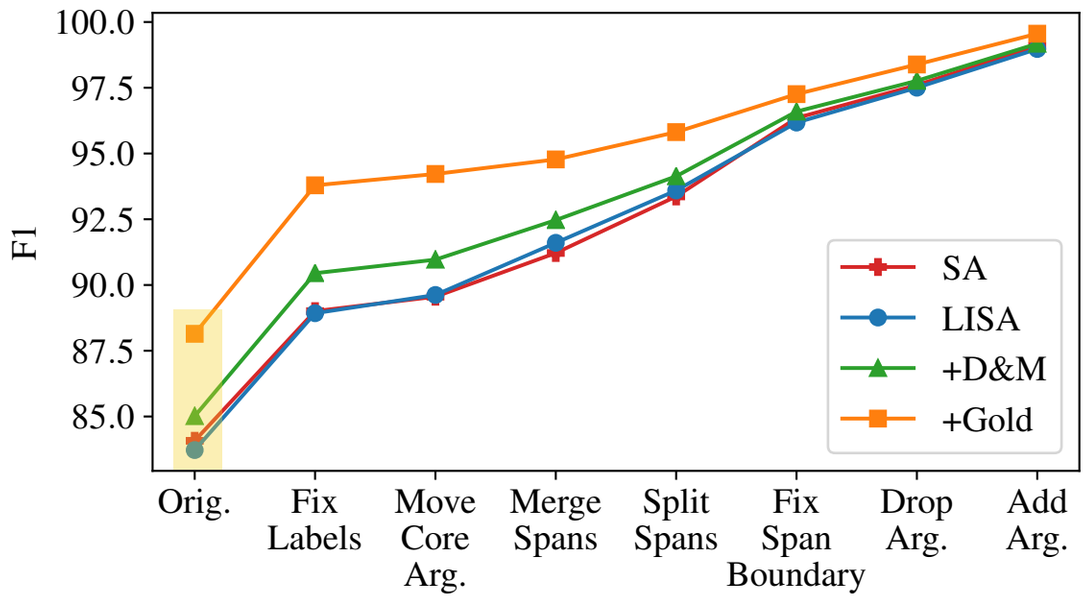
	GloVe	ELMo
	in-domain (dev)	in-domain (dev)

96.5 UAS!

Now I want to do a little analysis -- so I'm going to switch to development results --

1) and I want to show you how well the model does when provided with gold parses at test time.
2) With glove embeddings we see more than a 3 point improvement over our best model
3) and nearly 2.5 over our best ELMo model.
4) This is substantial considering the accuracy of the parser we're using, nearly 96.5 UAS! So, we did some more analysis to see what types of errors the gold parse is fixing.

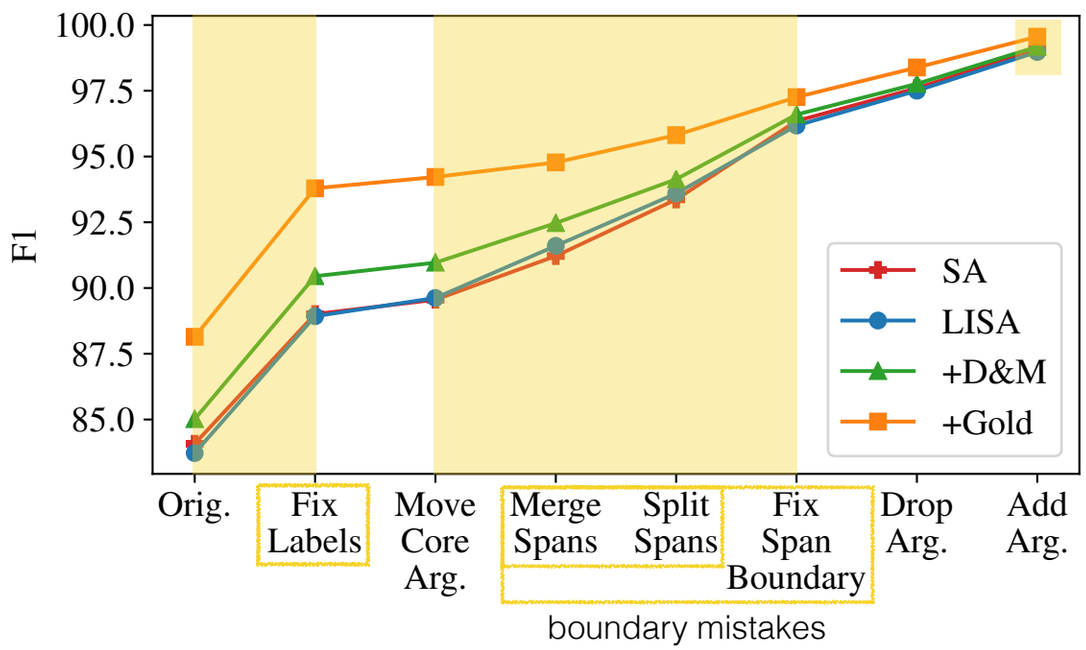
Experimental results: Analysis



Here we perform the same analysis as was originally presented by Luheng He and others in their 2017 paper:

1) starting from the model predictions, we incrementally fix common error types until

Experimental results: Analysis



1) (essentially) all of the errors are fixed
2) so one error type example is "fix labels," which corresponds to correcting the labels on spans whose boundaries were predicted correctly, and
3) another example is merging/splitting spans, which corresponds to either splitting a predicted span into two gold spans, or merging two adjacent predicted spans into a gold span.
4) First, we see that the most substantial source of errors across all models is from the labels on the spans: about 5 F1 points could be gained if we just fixed the labels on spans
5) Next, we see that using the gold parse helps the model mainly to identify span boundaries much better. This suggests that although the parse attachment score is high, its representation of spans is somehow really hurting the SRL model's ability to identify spans. Could be some interesting analysis for future work.

Summary

Thank you!

- **LISA**: Multi-task learning + multi-head self attention trained to attend to syntactic parents
  - Achieves state-of-the-art F1 on PropBank SRL
  - Linguistic structure improves generalization
  - Fast: encodes sequence *only once* to predict predicates, parts-of-speech, labeled dependency parse, SRL
- Everyone wants to run NLP on the entire web:
  - **accuracy**, **robustness**, **computational efficiency**.

Models & Code: `https://github.com/strubell/LISA`

I am on the academic job market this spring!

1) I've presented LISA: a model for multi-task learning where transfer between tasks is done through the attention mechanism
2) In this work we demonstrated that this works well for SRL but we think it could also help w/ other tasks across NLP
3) we've shown that injecting syntax helps even a really strong neural network model particularly when evaluating out-of-domain
4) We've observed in practice that this model is more efficient than alternatives and part of future work will be measuring this much more precisely
5) In conclusion, we believe this approach is the foundation for an NLP pipeline that will provide everyone w/ the accuracy, computational efficiency, and robustness that they need to run NLP on the entire web
6) Models and code are available online,
7) And that concludes my talk. Thanks for listening and I'm very happy to take questions!