Syntactically Supervised Transformers for Faster Neural Machine Translation

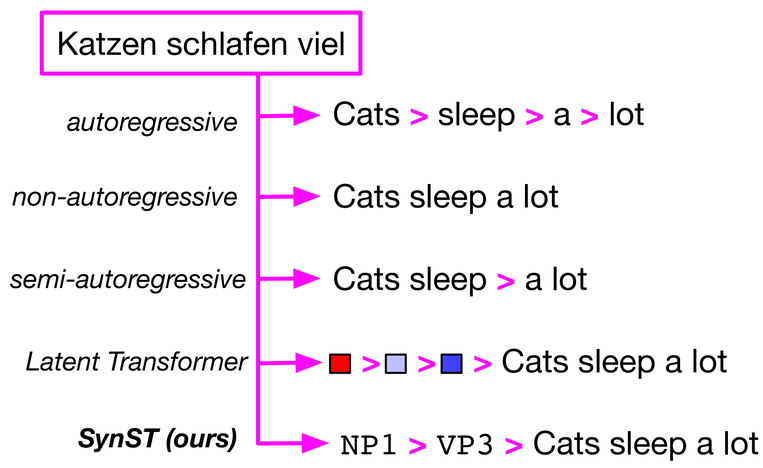
UMass Amherst

Nader Akoury, Kalpesh Krishna, Mohit Iyyer

code & data at github.com/dojoteef/synst

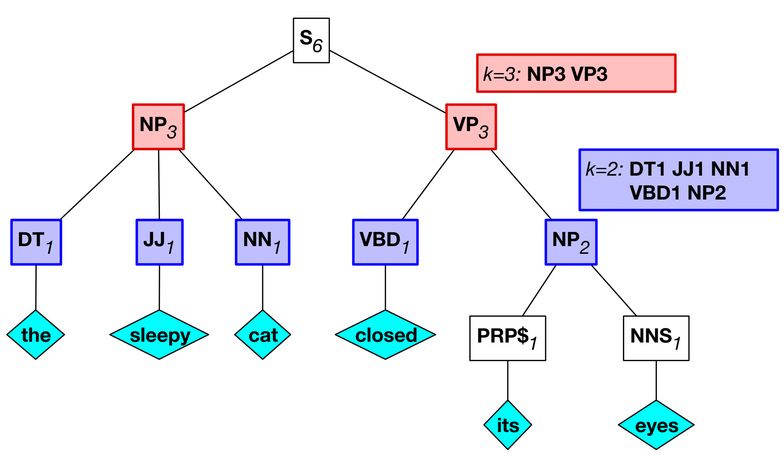
We present the syntactically supervised Transformer (**SynST**), which achieves faster translation and higher BLEU than competing non-autoregressive neural machine translation models.

SYNST VS. EXISTING SYSTEMS



Each > denotes a single decode step. Fewer decode steps results in faster translation, often at the expense of quality.

TARGET PARSE CHUNKING



During an in-order traversal, if the subtree rooted at a visited node spans ≤ *k* tokens, append it to our **chunk sequence**.

TRAINING SYNST

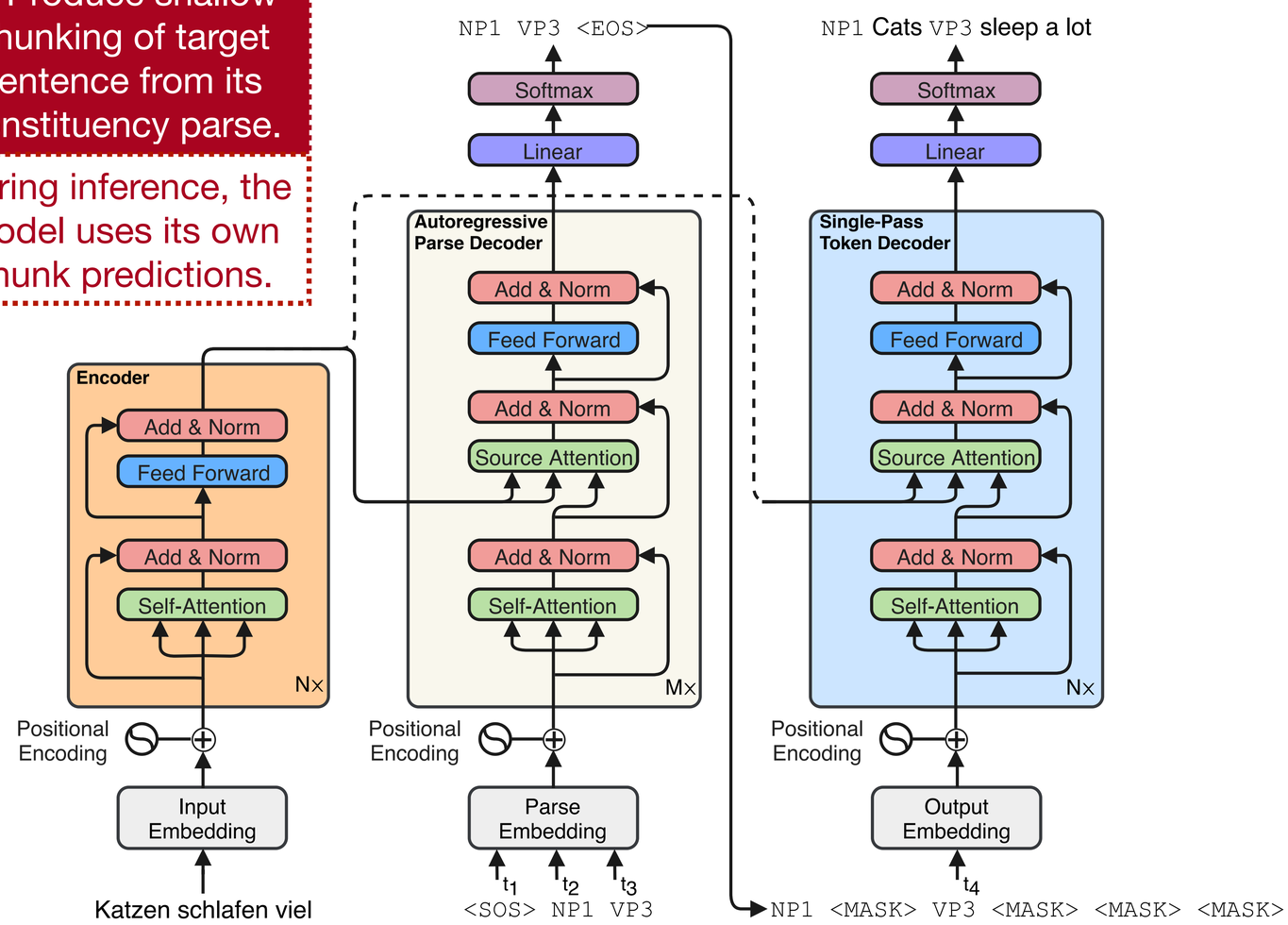
1. Encode source sentence.

2. Produce shallow chunking of target sentence from its constituency parse.

During inference, the model uses its own chunk predictions.

3. Autoregressively predict target chunks.

4. Non-autoregressively predict target tokens.



CONTROLLED EXPERIMENTS

Model	WMT En-De BLEU	WMT En-De Speedup	WMT De-En BLEU	WMT De-En Speedup	IWSLT En-De BLEU	IWSLT En-De Speedup	WMT En-Fr BLEU	WMT En-Fr Speedup
Vanilla Transformer								
Beam Size = 4	26.87	1.00×	30.73	1.00×	30.00	1.00×	40.22	1.00×
Beam Size = 1	25.82	1.15×	29.83	1.14×	28.66	1.16×	39.41	1.18×
Semi-Autoregressive Transformer								
k = 2	22.81	2.05×	26.78	2.04×	25.48	2.03×	36.62	2.14×
k = 4	16.44	3.61×	21.27	3.58×	20.25	3.45×	28.07	3.34×
k = 6	12.55	**4.86×**	15.23	4.27×	14.02	**4.39×**	24.63	4.77×
Latent Transformer*								
* As reported in (Kaiser et al. 2018)	19.8	3.89×	-	-	-	-	-	-
Syntactically Supervised Transformer								
k=6	20.74	**4.86×**	25.50	**5.06×**	23.82	3.78×	33.47	**5.32×**

ANALYSIS ON IWSLT DEV SET

Constituent identity is crucial for quality

Only predicting constituent length (`1` > `3`) rather than type & length (`NP1` > `VP3`), causes a **BLEU drop** from **23.8** to **8.2**.

Ground-truth syntax yields huge improvements

Conditioning on the ground-truth chunk sequence during inference dramatically **improves BLEU** from **23.8** to **41.5**, yielding an upper bound for our approach.

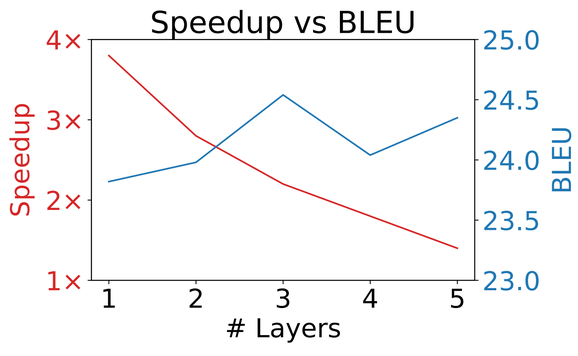
How much does SynST rely on syntax?

Source: *Katzen schlafen viel* Target: *Cats sleep a lot*
Predicted Parse: `NP1` > `VP2` Prediction: *Cats sleep lots*
Gold Parse: `NP1` > `VP3` Parsed Prediction: `NP1` > `VP2`

	Predicted Parse vs Gold Parse (trained separately)	Predicted Parse vs Gold Parse	Parsed Prediction vs Gold Parse	Parsed Prediction vs Predicted Parse
F1	65.48	69.64	79.16	89.90
Exact Match	4.23%	5.24%	5.94%	43.10%

Parsed prediction closely matches predicted parse, though there exists room for improvement for parse prediction.

SynST's bottleneck is its parse decoder



A one-layer parse decoder is **~3× faster** than a 5-layer version, with only a **~0.5 BLEU drop**.

Future work: dynamic vs fixed *k*

Randomly sampling possible chunk sequences during training by varying *k* leads to a **large BLEU improvement** (+1.5) with minimal impact to speedup (drop from 3.8× to 3.1×). Improving parse prediction is an avenue for future research.