Evolved Transformer

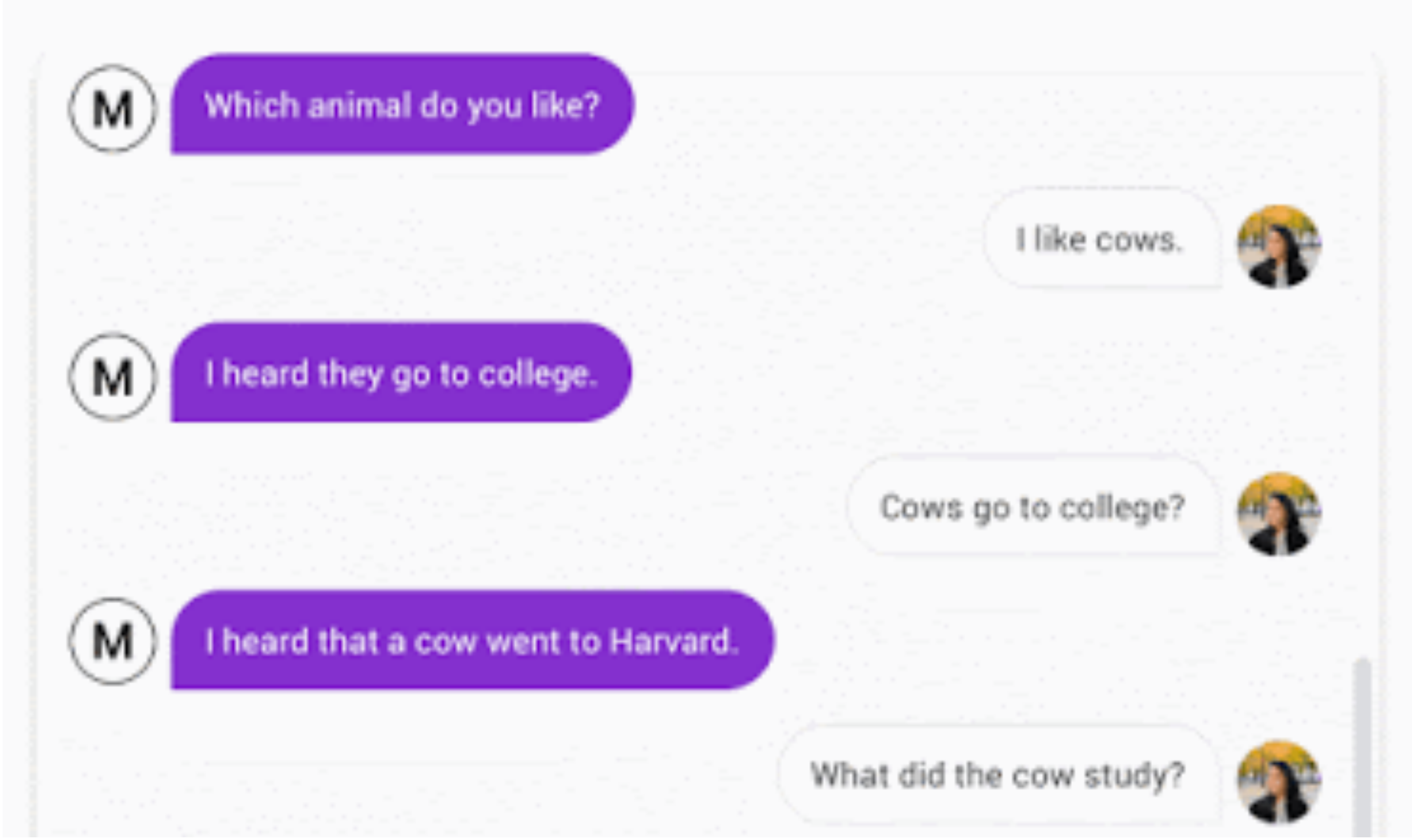
David R. So, Chen Liang, Quoc V. Le
Google Research, Brain Team
ICML 2019

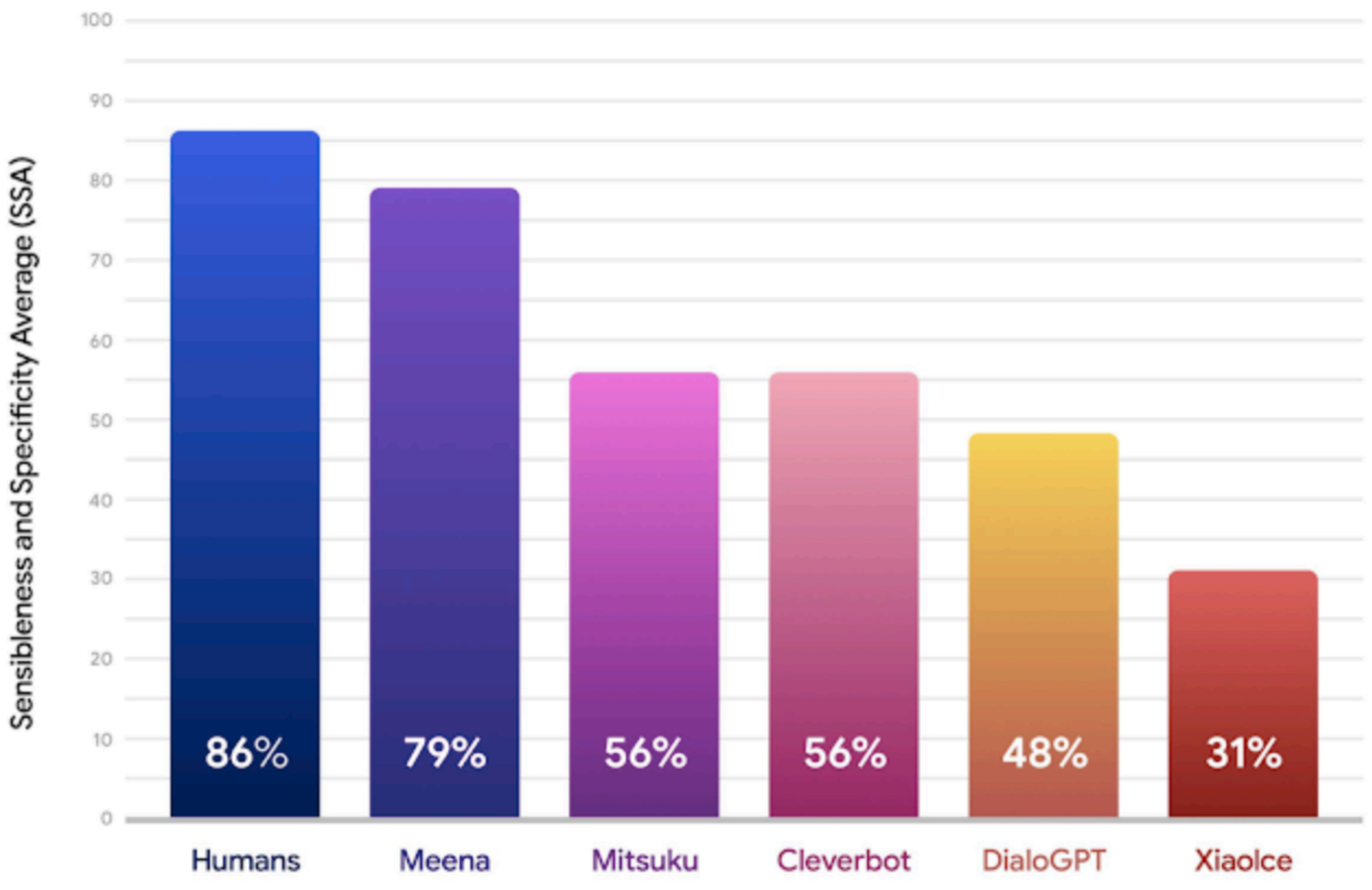
Kyung-Jae Cho VUNO Inc.

Towards a Conversational Agent that Can Chat About Anything

- Modern chatbots tend to be highly specialized
- Current open-domain chatbots often don't make sense
- Google Research proposed "human-like open domain chatbot"

Example of Meena encoding a 7-turn conversation context and generating a response, "The Next Generation".

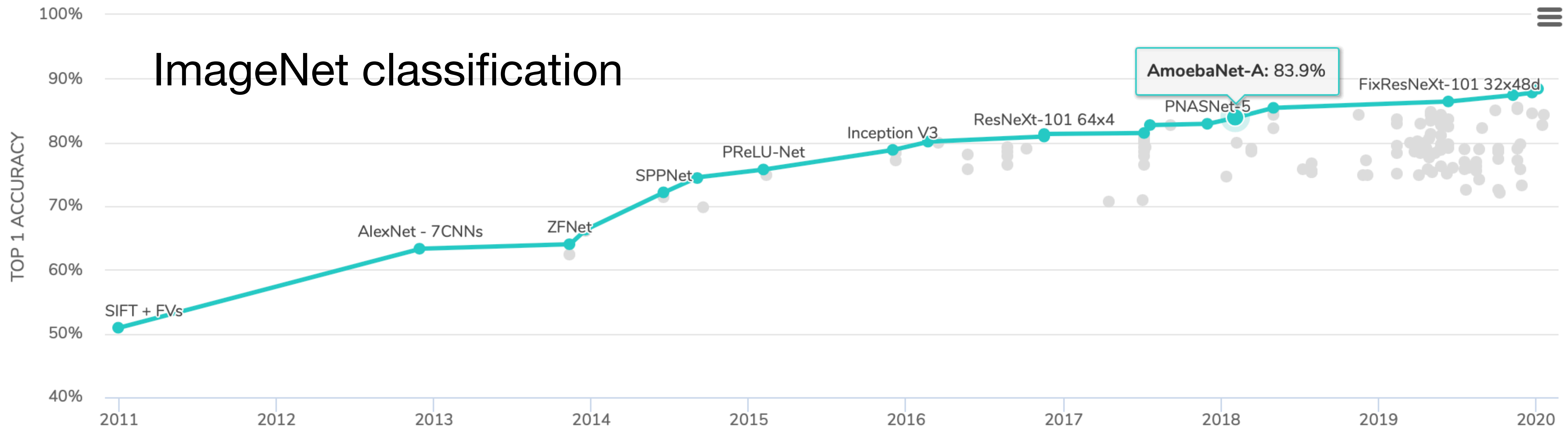




Meena Sensibleness and Specificity Average (SSA) compared with that of humans, Mitsuku, Cleverbot, XiaoIce, and DialoGPT.

Motivation

- **AutoML** produce models that exceed the perf. of those designed by humans (AmoebaNet, NasNet)
- The advances have mostly focused on improving **vision models**



- Some efforts has invested in searching for sequence models (NAS, ENAS)
  - However, they focused on improving RNNs
- Recent work shown better alternative (**transformer**) to RNNs on sequence problems

- **Goal: examine the use of NAS methods to design better transformer**

Related Work

- Field of NAS
  - Best architecture search methods are computationally intensive (AmoebaNet, NASNet)
  - Other methods developed speed in mind (DARTS, ENAS, SMASH .. etc)
  - Approach to both increase efficiency and search quality (PNAS, use of Hyperband)

- **This paper tries to use the most efficient and effective AutoML on searching the best transformer models**

Methods

- Employ evolution-based architecture search (tournament selection)
  - Simple & Efficient than reinforcement learning when resources are limited

- Proposed methods
  - Define search space to create new transformer architecture
  - Warm starting by seeding with Transformer
  - Proposed progressive dynamic hurdles (PDH)
    - Search on WMT 2014 English - German translation task is computationally expensive
    - Thus, need an efficient way search space

Search space

- NasNet-like search space
- Each cell contains blocks which receive two hidden state inputs and produce new hidden states as output
- The block perform separate transformation outputs to each input and combine the transformation output to produce a single block output
- Search space contains
  - **(1) five branch-level search fields**
  - **(2) one block-level search field**
  - **(3) one cell level search field**

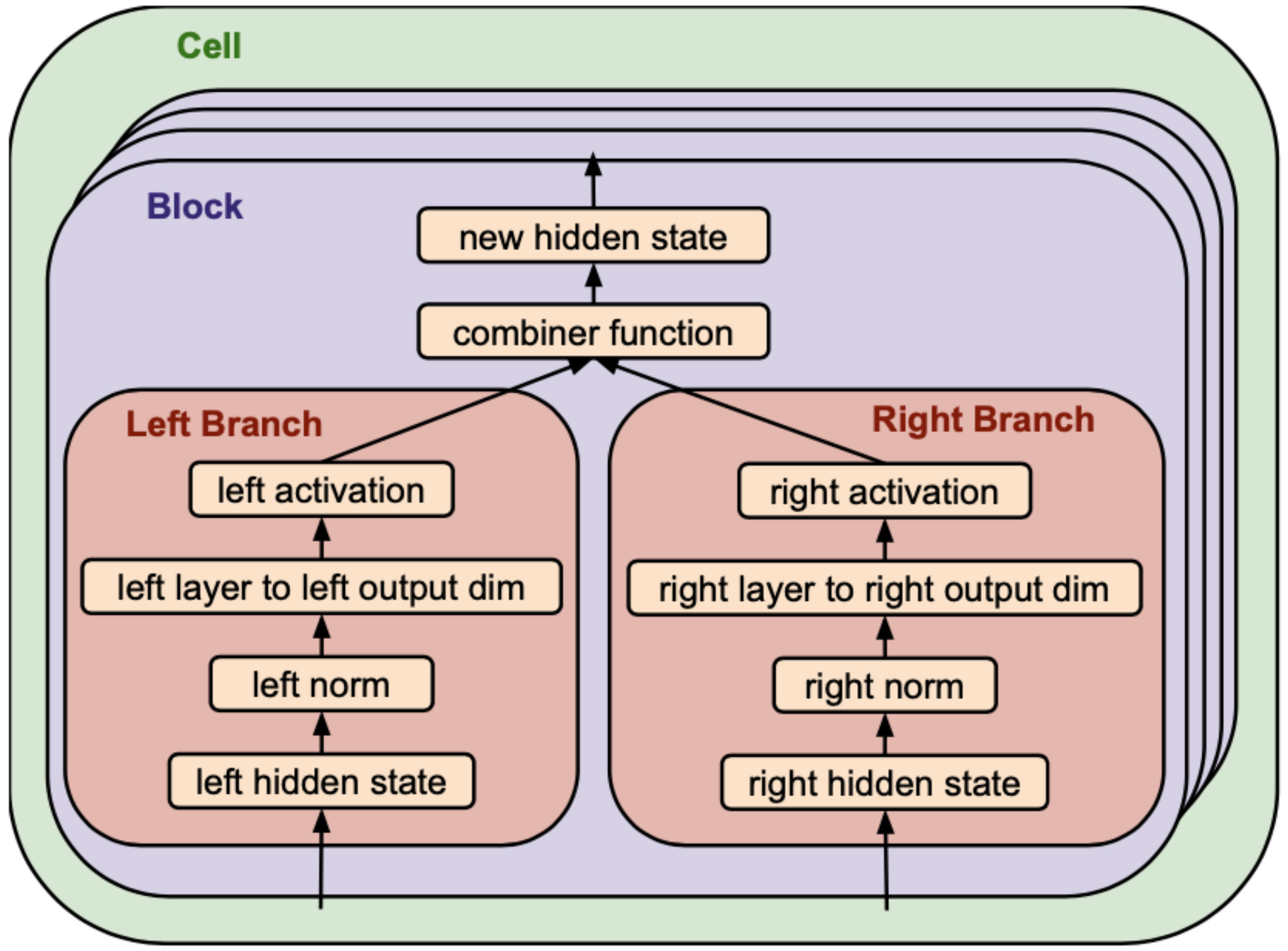


Figure 1. **Architecture composition from encoding.** Each block produces a new hidden state that is added to the pool of hidden states subsequent blocks can select as branch inputs. Each encoder has 6 unique blocks per cell and each decoder has 8 unique blocks per cell. Each cell is repeated *number of cells* times.

Five Branch-level search field

- **Input**
  - Specifies what hidden state from the block will be fed as input to the branch
  - For each i^{th} block, the branch input is [0,i)
- **Normalization**
  - [Layer Norm, None] applied to each input
- **Layers**
  - Neural network layer applied after normalization
- **Relative output dimension**
  - Used to specify the absolute output dimension
  - d: [1,10]
  - Every layer i, $a_i = d_i * s$, so that the parameter size is within a fixed range
- **Activations**
  - Non-linearity applied {SWISH, RELU, LEAKY RELU, None}

STANDARD CONV wx1: for $w \in \{1, 3\}$
DEPTHWISE SEPARABLE CONV wx1: for $w \in \{3, 5, 7, 9, 11\}$
LIGHTWEIGHT CONV wx1 r: for $w \in \{3, 5, 7, 15\}$, $r \in \{1, 4, 16\}$ (Wu et al., 2019). r is the reduction factor, equivalent to d/H described in Wu et al. (2019).
h HEAD ATTENTION: for $h \in \{4, 8, 16\}$
GATED LINEAR UNIT(Dauphin et al., 2017)
ATTEND TO ENCODER: (Only available to decoder)
IDENTITY: No transformation applied to input
DEAD BRANCH: No output

Block-level & Cell-level search field

- **Combiner functions**
  - {Addition, Concatenation, Multiplication}, padding applied if embedding depths are different
- **Number of cells**
  - [1,6]

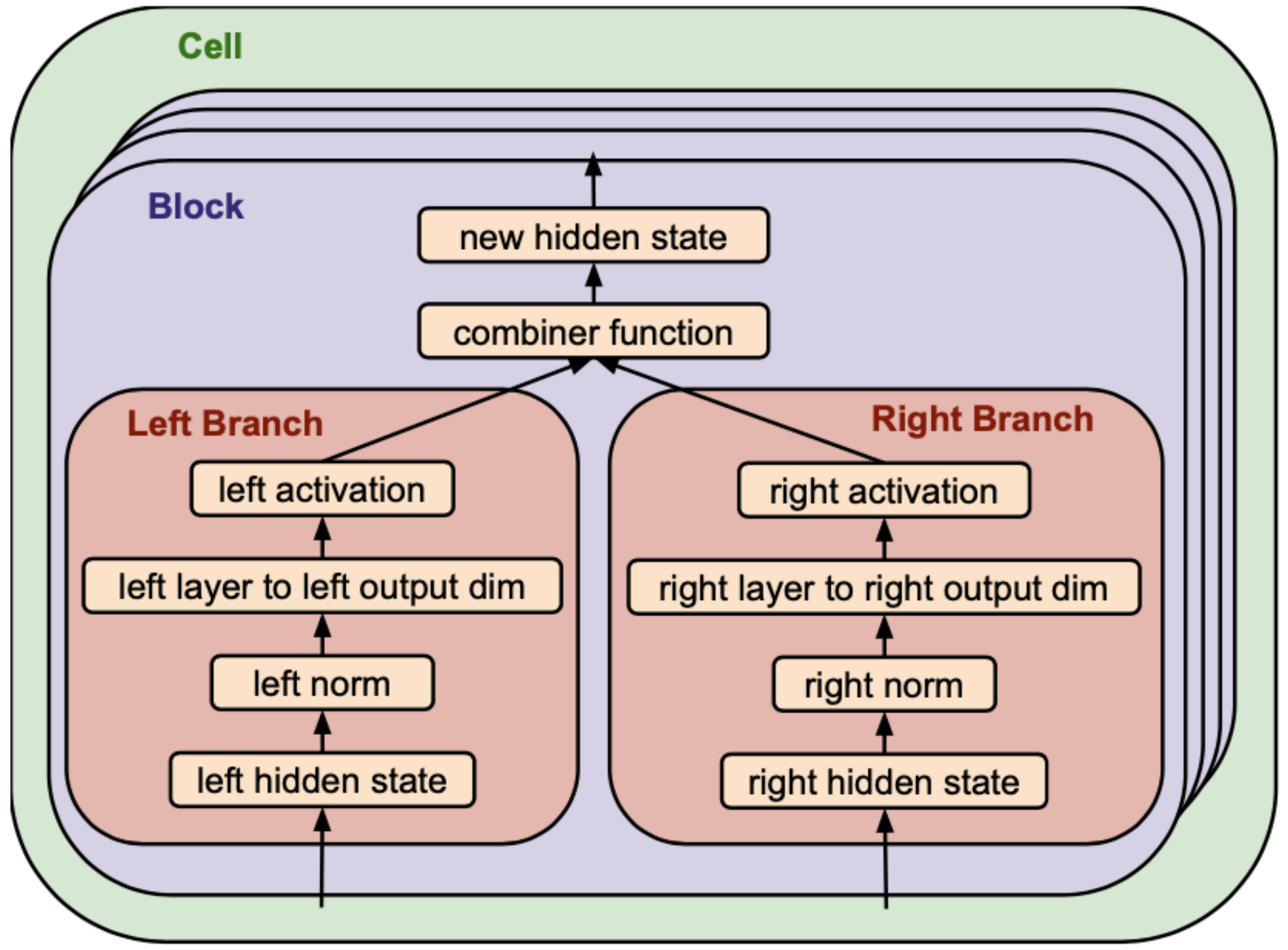


Figure 1. **Architecture composition from encoding.** Each block produces a new hidden state that is added to the pool of hidden states subsequent blocks can select as branch inputs. Each encoder has 6 unique blocks per cell and each decoder has 8 unique blocks per cell. Each cell is repeated *number of cells* times.

Tournament selection

- (1) Create population by creating P random models
- (2) Randomly sample models from population
- (3) Select parent with highest accuracy
- (4) Produce child by mutating the parent
- (5) Remove model with lowest accuracy in the population
- (6) Repeat (2) - (5) for C times

Algorithm 1 Aging Evolution

```
population ← empty queue                          ▷ The population.
history ← ∅                                       ▷ Will contain all models.
while |population| < P do                         ▷ Initialize population.
    model.arch ← RANDOMARCHITECTURE()
    model.accuracy ← TRAINANDEVAL(model.arch)
    add model to right of population
    add model to history
end while
while |history| < C do                            ▷ Evolve for C cycles.
    sample ← ∅                                    ▷ Parent candidates.
    while |sample| < S do
        candidate ← random element from population
                        ▷ The element stays in the population.
        add candidate to sample
    end while
    parent ← highest-accuracy model in sample
    child.arch ← MUTATE(parent.arch)
    child.accuracy ← TRAINANDEVAL(child.arch)
    add child to right of population
    add child to history
    remove dead from left of population           ▷ Oldest.
    discard dead
end while
return highest-accuracy model in history
```

Seeding the Search Space with Transformer

- To help navigate the large search space, we warm start the search process by seeding our initial population with a known strong model, original transformer
- This anchors the search to a known good starting point, and guarantees at least a single strong potential parent in the population
- The paper offer empirical support

Evolution with Progressive Dynamic Hurdles (PDH)

- WMT takes longer to train and evaluate
  - Takes 300K training steps (10 hours) using single TPU
- To address this problem we proposed PDH
  - Dynamically allocate resources to more promising architectures according to their fitness
- Hurdle -> mean fitness of current population

Algorithm 1 Calculate Model Fitness with Hurdles

inputs:
  $model$: the child model
  s: vector of train step increments
  h: queue of hurdles

append ∞ to h
TRAIN_N_STEPS($model$, s_0)
$fitness \leftarrow$ EVALUATE($model$)
$i \leftarrow 0$
$hurdle \leftarrow h_i$

while $fitness > hurdle$ **do**
  $i \leftarrow i + 1$
  TRAIN_N_STEPS($model$, s_i)
  $fitness \leftarrow$ EVALUATE($model$)
  $hurdle \leftarrow h_i$
end while
return $fitness$

Algorithm 2 Evolution Architecture Search with PDH

inputs:
  s: vector of train step increments
  m: number of child models per hurdle

$h \leftarrow$ *empty queue*
$i \leftarrow 0$
$population \leftarrow$ INITIAL_POPULATION()

while $i <$ LENGTH(s) - 1 **do**
  $population \leftarrow$ EVOL_N_MODELS($population$, m, s, h)
  $hurdle \leftarrow$ MEAN_FITNESS_OF_MAX($population$)
  append $hurdle$ to h
end while

$population \leftarrow$ EVOL_N_MODELS($population$, m, s, h)
return $population$

Experiment Setup

- Datasets
  - Machine Translation
    - WMT 18 En-De
    - WMT 14 En-Fr
    - WMT En-Cs
  - Language Modeling
    - 1 Billion Word Language Model Benchmark (LM1B)

- Training Details and Hyperparameters
  - Nearly identical to “Attention is all you need” settings
  - But modified to used memory-efficient Adafactor optimizer
  - Warm up to constant learning rate of 10-2 over 10k steps
  - Use inverse-square-root learning-rate decay
  - ..... etc

Ablation study of search techniques

- with PDH, with Transformer warm start
- Compared with equalize resource consumption

- Proposed search has best performance on average and lowest std
- Although "30K no hurdles" produced the best model but worst model at the same time
  - High standard deviation -> not stable
- Early stopping performed worse (15K vs. 30K)
- For 180K vs 300k it was resource inefficient and number of models were limited

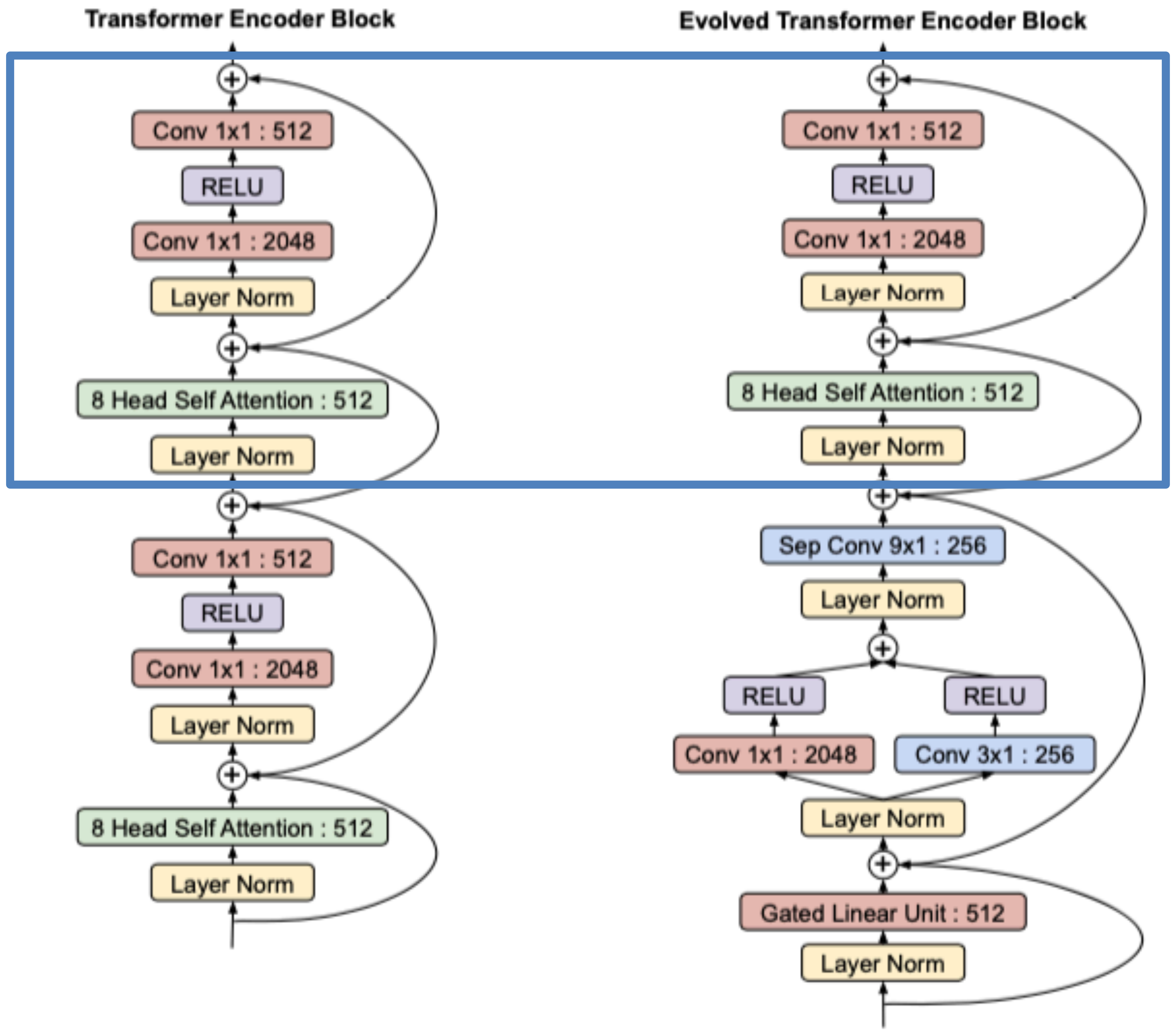
Seed Model	Train Steps	Num Models	Top Model Perplexity
Transformer	PDH	6000	**4.50** $\pm$ 0.01
Random	PDH	6000	5.23 $\pm$ 0.19
Transformer	15K	29714	4.57 $\pm$ 0.01
Transformer	30K	14857	4.53 $\pm$ 0.07
Transformer	180K	2477	4.58 $\pm$ 0.05
Transformer	300K	1486	4.61 $\pm$ 0.02

Table 1. **Top model validation perplexity of various search setups.** Number of models were chosen to equalize resource consumption.

Main search

- We launched a large scale version of our search
- The four most notable aspects of the found architecture
  - 1) wide depth-wise separable convolution
  - 2) Gated Linear Units
  - 3) Branching structures
  - 4) swish activations
- The latter portion is almost identical to the Transformer

Latter portion Identical



Performance and Analysis

- ET demonstrates stronger performance than the transformer at all sizes
- ET is more effective than the transformer at smaller model sizes

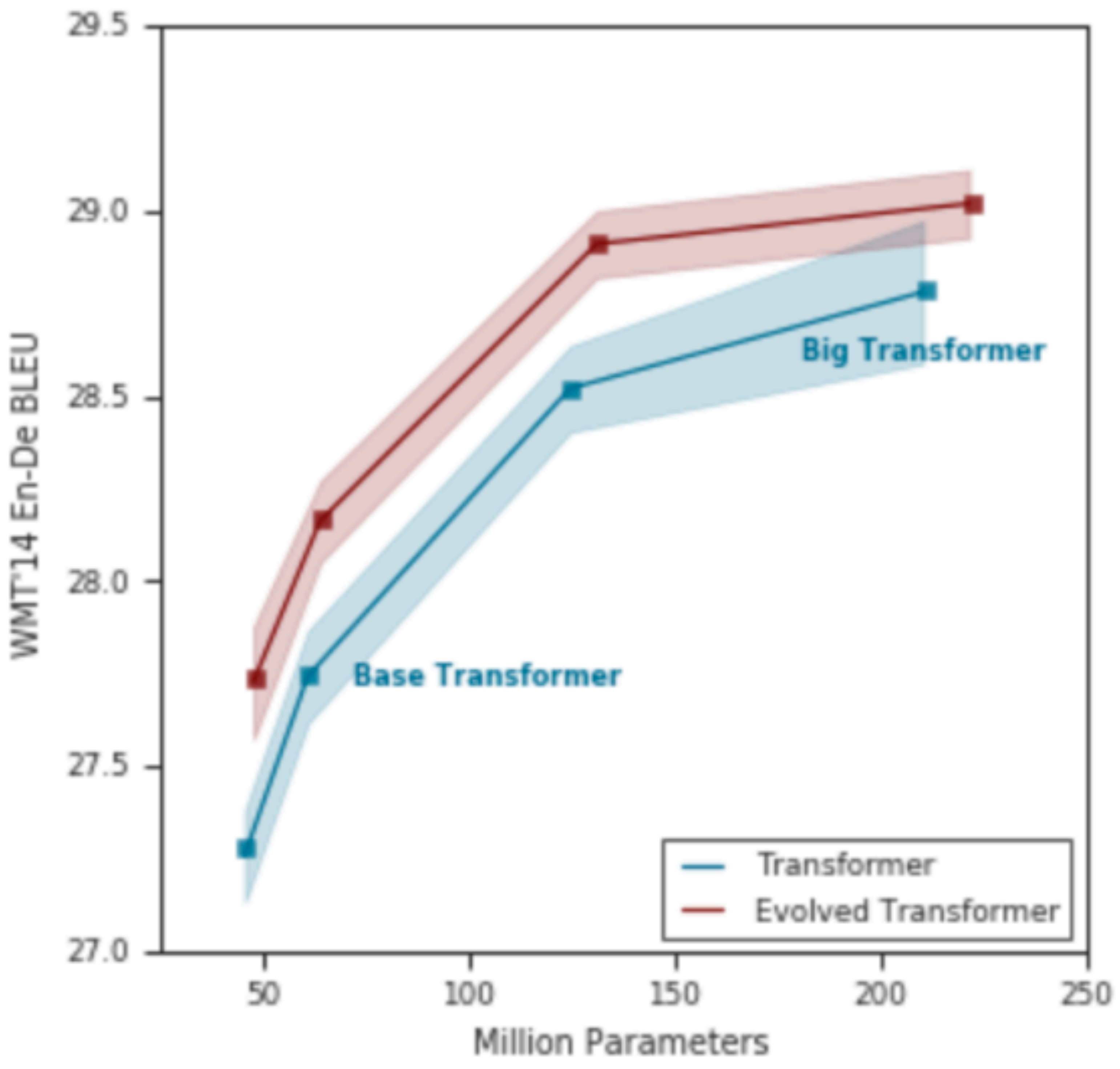


Figure 4. **Performance comparison of the Evolved Transformer against the Transformer across number of parameters.**

Generalization of Evolved Transformer

- Test if ET's strong performance generalizes
- At the "big" model size, BLEU performance saturates
  - Overfitting starts to occur at big model sizes
  - Data augmentation or hyperparameter tuning could improve performance

Task	Size	Tran Params	ET Params	Tran Perp	ET Perp	Tran BLEU	ET BLEU
WMT'14 En-De	Base	61.1M	64.1M	4.24 ± 0.03	$\mathbf{4.03} \pm 0.02$	28.2 ± 0.2	$\mathbf{28.4} \pm 0.2$
WMT'14 En-De	Big	210.4M	221.7M	3.87 ± 0.02	$\mathbf{3.77} \pm 0.02$	29.1 ± 0.1	$\mathbf{29.3} \pm 0.1$
WMT'14 En-De	Deep	224.0M	218.1M	3.86 ± 0.02	$\mathbf{3.69} \pm 0.01$	29.2 ± 0.1	$\mathbf{29.5} \pm 0.1$
WMT'14 En-Fr	Base	60.8	63.8M	3.61 ± 0.01	$\mathbf{3.42} \pm 0.01$	40.0 ± 0.1	$\mathbf{40.6} \pm 0.1$
WMT'14 En-Fr	Big	209.8M	221.2M	3.26 ± 0.01	$\mathbf{3.13} \pm 0.01$	41.2 ± 0.1	$\mathbf{41.3} \pm 0.1$
WMT'14 En-Cs	Base	59.8M	62.7M	4.98 ± 0.04	$\mathbf{4.42} \pm 0.01$	27.0 ± 0.1	$\mathbf{27.6} \pm 0.2$
WMT'14 En-Cs	Big	207.6M	218.9M	4.43 ± 0.01	$\mathbf{4.38} \pm 0.03$	28.1 ± 0.1	$\mathbf{28.2} \pm 0.1$
LM1B	Big	141.1M	151.8M	30.44 ± 0.04	$\mathbf{28.60} \pm 0.03$	-	-

Compare with other previous results

- Evolved Transformer achieved a new SOTA

Model	Params	BLEU	SacreBLEU (Post, 2018)
Gehring et al. (2017)	216M	25.2	-
Vaswani et al. (2017)	213M	28.4	-
Ahmed et al. (2017)	213M	28.9	-
Chen et al. (2018)	379M	28.5	-
Shaw et al. (2018)	213M	29.2	-
Ott et al. (2018)	210M	29.3	28.6
Wu et al. (2019)	213M	29.7	-
Evolved Transformer	218M	**29.8**	**29.2**

Table 4. **Model comparison on WMT'14 En-De.**

Conclusion

- Presented first neural architecture search conducted to find improved transformer
- To mitigate the size of the search space and the cost of training child models, the paper proposed progressive dynamic hurdles method and warm starting
- In experiment the Evolved Transformer showed consistent stronger performance on both translation and language modeling
- On WMT 14 En-De, the ET established new SOTA of 29.8 BLEU
- It also proved to be efficient at smaller sizes achieving the same quality as the original “big” transformer with 37.6% less parameters

Putting the world’s medical data to work

kcho035@vuno.co