<p><br></p><p>Google Research</p><p>nikip@google.com</p><p>Jakob Uszkoreit</p><p>∗</p><p>Google Research</p><p>usz@google.com</p><p>Llion Jones</p><p>∗</p><p>Google Research</p><p>llion@google.com</p><p>Aidan N. Gomez</p><p>∗ †</p><p>University of Toronto</p><p>aidan@cs.toronto.edu</p><p>Łukasz Kaiser</p><p>∗</p><p>Google Brain</p><p>lukaszkaiser@google.com</p><p>Illia Polosukhin</p><p>∗ ‡</p><p>illia.polosukhin@gmail.com</p><p>Abstract</p><p>The dominant sequence transduction models are based on complex recurrent or</p><p>convolutional neural networks that include an encoder and a decoder. The best</p><p>performing models also connect the encoder and decoder through an attention</p><p>mechanism. We propose a new simple network architecture, the Transformer,</p><p>based solely on attention mechanisms, dispensing with recurrence and convolutions</p><p>entirely. Experiments on two machine translation tasks show these models to</p><p>be superior in quality while being more parallelizable and requiring significantly</p><p>less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-</p><p>to-German translation task, improving over the existing best results, including</p><p>ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task,</p><p>our model establishes a new single-model state-of-the-art BLEU score of 41.8 after</p><p>training for 3.5 days on eight GPUs, a small fraction of the training costs of the</p><p>best models from the literature. We show that the Transformer generalizes well to</p><p>other tasks by applying it successfully to English constituency parsing both with</p><p>large and limited training data.</p><p>∗</p><p>Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started</p><p>the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and</p><p>has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head</p><p>attention and the parameter-free position representation and became the other person involved in nearly every</p><p>detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and</p><p>tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and</p><p>efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and</p><p>implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating</p><p>our research.</p><p>†</p><p>Work performed while at Google Brain.</p><p>‡</p><p>Work performed while at Google Research.</p><p>31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.</p><p>arXiv:1706.03762v7 [cs.CL] 2 Aug 2023</p><p>1 Introduction</p><p>Recurrent neural networks, long short-term memory [13] and gated recurrent [7] neural networks</p><p>in particular, have been firmly established as state of the art approaches in sequence modeling and</p><p>transduction problems such as language modeling and machine translation [35,2,5]. Numerous</p><p>efforts have since continued to push the boundaries of recurrent language models and encoder-decoder</p><p>architectures [38, 24, 15].</p><p>Recurrent models typically factor computation along the symbol positions of the input and output</p><p>sequences. Aligning the positions to steps in computation time, they generate a sequence of hidden</p><p>statesh</p><p>t</p><p>, as a function of the previous hidden stateh</p><p>t−1</p><p>and the input for positiont. This inherently</p><p>sequential nature precludes parallelization within training examples, which becomes critical at longer</p><p>sequence lengths, as memory constraints limit batching across examples. Recent work has achieved</p><p>significant improvements in computational efficiency through factorization tricks [21] and conditional</p><p>computation [32], while also improving model performance in case of the latter. The fundamental</p><p>constraint of sequential computation, however, remains.</p><p>Attention mechanisms have become an integral part of compelling sequence modeling and transduc-</p><p>tion models in various tasks, allowing modeling of dependencies without regard to their distance in</p><p>the input or output sequences [2,19]. In all but a few cases [27], however, such attention mechanisms</p><p>are used in conjunction with a recurrent network.</p><p>In this work we propose the Transformer, a model architecture eschewing recurrence and instead</p><p>relying entirely on an attention mechanism to draw global dependencies between input and output.</p><p>The Transformer allows for significantly more parallelization and can reach a new state of the art in</p><p>translation quality after being trained for as little as twelve hours on eight P100 GPUs.</p><p>2 Background</p><p>The goal of reducing sequential computation also forms the foundation of the Extended Neural GPU</p><p>[16], ByteNet [18] and ConvS2S [9], all of which use convolutional neural networks as basic building</p><p>block, computing hidden representations in parallel for all input and output positions. In these models,</p><p>the number of operations required to relate signals from two arbitrary input or output positions grows</p><p>in the distance between positions, linearly for ConvS2S and logarithmically for ByteNet. This makes</p><p>it more difficult to learn dependencies between distant positions [12]. In the Transformer this is</p><p>reduced to a constant number of operations, albeit at the cost of reduced effective resolution due</p><p>to averaging attention-weighted positions, an effect we counteract with Multi-Head Attention as</p><p>described in section 3.2.</p><p>Self-attention, sometimes called intra-attention is an attention mechanism relating different positions</p><p>of a single sequence in order to compute a representation of the sequence. Self-attention has been</p><p>used successfully in a variety of tasks including reading comprehension, abstractive summarization,</p><p>textual entailment and learning task-independent sentence representations [4, 27, 28, 22].</p><p>End-to-end memory networks are based on a recurrent attention mechanism instead of sequence-</p><p>aligned recurrence and have been shown to perform well on simple-language question answering and</p><p>language modeling tasks [34].</p><p>To the best of our knowledge, however, the Transformer is the first transduction model relying</p><p>entirely on self-attention to compute representations of its input and output without using sequence-</p><p>aligned RNNs or convolution. In the following sections, we will describe the Transformer, motivate</p><p>self-attention and discuss its advantages over models such as [17, 18] and [9].</p><p>3 Model Architecture</p><p>Most competitive neural sequence transduction models have an encoder-decoder structure [5,2,35].</p><p>Here, the encoder maps an input sequence of symbol representations(x</p><p>1</p><p>,...,x</p><p>n</p><p>)to a sequence</p><p>of continuous representationsz= (z</p><p>1</p><p>,...,z</p><p>n</p><p>). Givenz, the decoder then generates an output</p><p>sequence(y</p><p>1</p><p>,...,y</p><p>m</p><p>)of symbols one element at a time. At each step the model is auto-regressive</p><p>[10], consuming the previously generated symbols as additional input when generating the next.</p><p>2</p><p>Figure 1: The Transformer - model architecture.</p><p>The Transformer follows this overall architecture using stacked self-attention and point-wise, fully</p><p>connected layers for both the encoder and decoder, shown in the left and right halves of Figure 1,</p><p>respectively.</p><p>3.1 Encoder and Decoder Stacks</p><p>Encoder:The encoder is composed of a stack ofN= 6identical layers. Each layer has two</p><p>sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-</p><p>wise fully connected feed-forward network. We employ a residual connection [11] around each of</p><p>the two sub-layers, followed by layer normalization [1]. That is, the output of each sub-layer is</p><p>LayerNorm(x+ Sublayer(x)), whereSublayer(x)is the function implemented by the sub-layer</p><p>itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding</p><p>layers, produce outputs of dimensiond</p><p>model</p><p>= 512.</p><p>Decoder:The decoder is also composed of a stack ofN= 6identical layers. In addition to the two</p><p>sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head</p><p>attention over the output of the encoder stack. Similar to the encoder, we employ residual connections</p><p>around each of the sub-layers, followed by layer normalization. We also modify the self-attention</p><p>sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This</p><p>masking, combined with fact that the output embeddings are offset by one position, ensures that the</p><p>predictions for positionican depend only on the known outputs at positions less thani.</p><p>3.2 Attention</p><p>An attention function can be described as mapping a query and a set of key-value pairs to an output,</p><p>where the query, keys, values, and output are all vectors. The output is computed as a weighted sum</p><p>3</p><p>Scaled Dot-Product AttentionMulti-Head Attention</p><p>Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several</p><p>attention layers running in parallel.</p><p>of the values, where the weight assigned to each value is computed by a compatibility function of the</p><p>query with the corresponding key.</p><p>3.2.1 Scaled Dot-Product Attention</p><p>We call our particular attention "Scaled Dot-Product Attention" (Figure 2). The input consists of</p><p>queries and keys of dimensiond</p><p>k</p><p>, and values of dimensiond</p><p>v</p><p>. We compute the dot products of the</p><p>query with all keys, divide each by</p><p>√</p><p>d</p><p>k</p><p>, and apply a softmax function to obtain the weights on the</p><p>values.</p><p>In practice, we compute the attention function on a set of queries simultaneously, packed together</p><p>into a matrixQ. The keys and values are also packed together into matricesKandV. We compute</p><p>the matrix of outputs as:</p><p>Attention(Q,K,V) = softmax(</p><p>QK</p><p>T</p><p>√</p><p>d</p><p>k</p><p>)V(1)</p><p>The two most commonly used attention functions are additive attention [2], and dot-product (multi-</p><p>plicative) attention. Dot-product attention is identical to our algorithm, except for the scaling factor</p><p>of</p><p>1</p><p>√</p><p>d</p><p>k</p><p>. Additive attention computes the compatibility function using a feed-forward network with</p><p>a single hidden layer. While the two are similar in theoretical complexity, dot-product attention is</p><p>much faster and more space-efficient in practice, since it can be implemented using highly optimized</p><p>matrix multiplication code.</p><p>While for small values ofd</p><p>k</p><p>the two mechanisms perform similarly, additive attention outperforms</p><p>dot product attention without scaling for larger values ofd</p><p>k</p><p>[3]. We suspect that for large values of</p><p>d</p><p>k</p><p>, the dot products grow large in magnitude, pushing the softmax function into regions where it has</p><p>extremely small gradients</p><p>4</p><p>. To counteract this effect, we scale the dot products by</p><p>1</p><p>√</p><p>d</p><p>k</p><p>.</p><p>3.2.2 Multi-Head Attention</p><p>Instead of performing a single attention function withd</p><p>model</p><p>-dimensional keys, values and queries,</p><p>we found it beneficial to linearly project the queries, keys and valueshtimes with different, learned</p><p>linear projections tod</p><p>k</p><p>,d</p><p>k</p><p>andd</p><p>v</p><p>dimensions, respectively. On each of these projected versions of</p><p>queries, keys and values we then perform the attention function in parallel, yieldingd</p><p>v</p><p>-dimensional</p><p>4</p><p>To illustrate why the dot products get large, assume that the components ofqandkare independent random</p><p>variables with mean0and variance1. Then their dot product,q·k=</p><p>P</p><p>d</p><p>k</p><p>i=1</p><p>q</p><p>i</p><p>k</p><p>i</p><p>, has mean0and varianced</p><p>k</p><p>.</p><p>4</p><p>output values. These are concatenated and once again projected, resulting in the final values, as</p><p>depicted in Figure 2.</p><p>Multi-head attention allows the model to jointly attend to information from different representation</p><p>subspaces at different positions. With a single attention head, averaging inhibits this.</p><p>MultiHead(Q,K,V) = Concat(head</p><p>1</p><p>,...,head</p><p>h</p><p>)W</p><p>O</p><p>wherehead</p><p>i</p><p>= Attention(QW</p><p>Q</p><p>i</p><p>,KW</p><p>K</p><p>i</p><p>,V W</p><p>V</p><p>i</p><p>)</p><p>Where the projections are parameter matricesW</p><p>Q</p><p>i</p><p>∈R</p><p>d</p><p>model</p><p>×d</p><p>k</p><p>,W</p><p>K</p><p>i</p><p>∈R</p><p>d</p><p>model</p><p>×d</p><p>k</p><p>,W</p><p>V</p><p>i</p><p>∈R</p><p>d</p><p>model</p><p>×d</p><p>v</p><p>andW</p><p>O</p><p>∈R</p><p>hd</p><p>v</p><p>×d</p><p>model</p><p>.</p><p>In this work we employh= 8parallel attention layers, or heads. For each of these we use</p><p>d</p><p>k</p><p>=d</p><p>v</p><p>=d</p><p>model</p><p>/h= 64. Due to the reduced dimension of each head, the total computational cost</p><p>is similar to that of single-head attention with full dimensionality.</p><p>3.2.3 Applications of Attention in our Model</p><p>The Transformer uses multi-head attention in three different ways:</p><p>•</p><p>In "encoder-decoder attention" layers, the queries come from the previous decoder layer,</p><p>and the memory keys and values come from the output of the encoder. This allows every</p><p>position in the decoder to attend over all positions in the input sequence. This mimics the</p><p>typical encoder-decoder attention mechanisms in sequence-to-sequence models such as</p><p>[38, 2, 9].</p><p>•The encoder contains self-attention layers. In a self-attention layer all of the keys, values</p><p>and queries come from the same place, in this case, the output of the previous layer in the</p><p>encoder. Each position in the encoder can attend to all positions in the previous layer of the</p><p>encoder.</p><p>•Similarly, self-attention layers in the decoder allow each position in the decoder to attend to</p><p>all positions in the decoder up to and including that position. We need to prevent leftward</p><p>information flow in the decoder to preserve the auto-regressive property. We implement this</p><p>inside of scaled dot-product attention by masking out (setting to−∞) all values in the input</p><p>of the softmax which correspond to illegal connections. See Figure 2.</p><p>3.3 Position-wise Feed-Forward Networks</p><p>In addition to attention sub-layers, each of the layers in our encoder and decoder contains a fully</p><p>connected feed-forward network, which is applied to each position separately and identically. This</p><p>consists of two linear transformations with a ReLU activation in between.</p><p>FFN(x) = max(0,xW</p><p>1</p><p>+b</p><p>1</p><p>)W</p><p>2</p><p>+b</p><p>2</p><p>(2)</p><p>While the linear transformations are the same across different positions, they use different parameters</p><p>from layer to layer. Another way of describing this is as two convolutions with kernel size 1.</p><p>The dimensionality of input and output isd</p><p>model</p><p>= 512, and the inner-layer has dimensionality</p><p>d</p><p>ff</p><p>= 2048.</p><p>3.4 Embeddings and Softmax</p><p>Similarly to other sequence transduction models, we use learned embeddings to convert the input</p><p>tokens and output tokens to vectors of dimensiond</p><p>model</p><p>. We also use the usual learned linear transfor-</p><p>mation and softmax function to convert the decoder output to predicted next-token probabilities. In</p><p>our model, we share the same weight matrix between the two embedding layers and the pre-softmax</p><p>linear transformation, similar to [30]. In the embedding layers, we multiply those weights by</p><p>√</p><p>d</p><p>model</p><p>.</p><p>5</p><p>Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations</p><p>for different layer types.nis the sequence length,dis the representation dimension,kis the kernel</p><p>size of convolutions andrthe size of the neighborhood in restricted self-attention.</p><p>Layer TypeComplexity per LayerSequentialMaximum Path Length</p><p>Operations</p><p>Self-AttentionO(n</p><p>2</p><p>·d)O(1)O(1)</p><p>RecurrentO(n·d</p><p>2</p><p>)O(n)O(n)</p><p>ConvolutionalO(k·n·d</p><p>2</p><p>)O(1)O(log</p><p>k</p><p>(n))</p><p>Self-Attention (restricted)O(r·n·d)O(1)O(n/r)</p><p>3.5 Positional Encoding</p><p>Since our model contains no recurrence and no convolution, in order for the model to make use of the</p><p>order of the sequence, we must inject some information about the relative or absolute position of the</p><p>tokens in the sequence. To this end, we add "positional encodings" to the input embeddings at the</p><p>bottoms of the encoder and decoder stacks. The positional encodings have the same dimensiond</p><p>model</p><p>as the embeddings, so that the two can be summed. There are many choices of positional encodings,</p><p>learned and fixed [9].</p><p>In this work, we use sine and cosine functions of different frequencies:</p><p>PE</p><p>(pos,2i)</p><p>=sin(pos/10000</p><p>2i/d</p><p>model</p><p>)</p><p>PE</p><p>(pos,2i+1)</p><p>=cos(pos/10000</p><p>2i/d</p><p>model</p><p>)</p><p>whereposis the position andiis the dimension. That is, each dimension of the positional encoding</p><p>corresponds to a sinusoid. The wavelengths form a geometric progression from2πto10000·2π. We</p><p>chose this function because we hypothesized it would allow the model to easily learn to attend by</p><p>relative positions, since for any fixed offsetk,PE</p><p>pos+k</p><p>can be represented as a linear function of</p><p>PE</p><p>pos</p><p>.</p><p>We also experimented with using learned positional embeddings [9] instead, and found that the two</p><p>versions produced nearly identical results (see Table 3 row (E)). We chose the sinusoidal version</p><p>because it may allow the model to extrapolate to sequence lengths longer than the ones encountered</p><p>during training.</p><p>4 Why Self-Attention</p><p>In this section we compare various aspects of self-attention layers to the recurrent and convolu-</p><p>tional layers commonly used for mapping one variable-length sequence of symbol representations</p><p>(x</p><p>1</p><p>,...,x</p><p>n</p><p>)to another sequence of equal length(z</p><p>1</p><p>,...,z</p><p>n</p><p>), withx</p><p>i</p><p>,z</p><p>i</p><p>∈R</p><p>d</p><p>, such as a hidden</p><p>layer in a typical sequence transduction encoder or decoder. Motivating our use of self-attention we</p><p>consider three desiderata.</p><p>One is the total computational complexity per layer. Another is the amount of computation that can</p><p>be parallelized, as measured by the minimum number of sequential operations required.</p><p>The third is the path length between long-range dependencies in the network. Learning long-range</p><p>dependencies is a key challenge in many sequence transduction tasks. One key factor affecting the</p><p>ability to learn such dependencies is the length of the paths forward and backward signals have to</p><p>traverse in the network. The shorter these paths between any combination of positions in the input</p><p>and output sequences, the easier it is to learn long-range dependencies [12]. Hence we also compare</p><p>the maximum path length between any two input and output positions in networks composed of the</p><p>different layer types.</p><p>As noted in Table 1, a self-attention layer connects all positions with a constant number of sequentially</p><p>executed operations, whereas a recurrent layer requiresO(n)sequential operations. In terms of</p><p>computational complexity, self-attention layers are faster than recurrent layers when the sequence</p><p>6</p><p>lengthnis smaller than the representation dimensionalityd, which is most often the case with</p><p>sentence representations used by state-of-the-art models in machine translations, such as word-piece</p><p>[38] and byte-pair [31] representations. To improve computational performance for tasks involving</p><p>very long sequences, self-attention could be restricted to considering only a neighborhood of sizerin</p><p>the input sequence centered around the respective output position. This would increase the maximum</p><p>path length toO(n/r). We plan to investigate this approach further in future work.</p><p>A single convolutional layer with kernel widthk &lt; ndoes not connect all pairs of input and output</p><p>positions. Doing so requires a stack ofO(n/k)convolutional layers in the case of contiguous kernels,</p><p>orO(log</p><p>k</p><p>(n))in the case of dilated convolutions [18], increasing the length of the longest paths</p><p>between any two positions in the network. Convolutional layers are generally more expensive than</p><p>recurrent layers, by a factor ofk. Separable convolutions [6], however, decrease the complexity</p><p>considerably, toO(k·n·d+n·d</p><p>2</p><p>). Even withk=n, however, the complexity of a separable</p><p>convolution is equal to the combination of a self-attention layer and a point-wise feed-forward layer,</p><p>the approach we take in our model.</p><p>As side benefit, self-attention could yield more interpretable models. We inspect attention distributions</p><p>from our models and present and discuss examples in the appendix. Not only do individual attention</p><p>heads clearly learn to perform different tasks, many appear to exhibit behavior related to the syntactic</p><p>and semantic structure of the sentences.</p><p>5 Training</p><p>This section describes the training regime for our models.</p><p>5.1 Training Data and Batching</p><p>We trained on the standard WMT 2014 English-German dataset consisting of about 4.5 million</p><p>sentence pairs. Sentences were encoded using byte-pair encoding [3], which has a shared source-</p><p>target vocabulary of about 37000 tokens. For English-French, we used the significantly larger WMT</p><p>2014 English-French dataset consisting of 36M sentences and split tokens into a 32000 word-piece</p><p>vocabulary [38]. Sentence pairs were batched together by approximate sequence length. Each training</p><p>batch contained a set of sentence pairs containing approximately 25000 source tokens and 25000</p><p>target tokens.</p><p>5.2 Hardware and Schedule</p><p>We trained our models on one machine with 8 NVIDIA P100 GPUs. For our base models using</p><p>the hyperparameters described throughout the paper, each training step took about 0.4 seconds. We</p><p>trained the base models for a total of 100,000 steps or 12 hours. For our big models,(described on the</p><p>bottom line of table 3), step time was 1.0 seconds. The big models were trained for 300,000 steps</p><p>(3.5 days).</p><p>5.3 Optimizer</p><p>We used the Adam optimizer [20] withβ</p><p>1</p><p>= 0.9,β</p><p>2</p><p>= 0.98andε= 10</p><p>−9</p><p>. We varied the learning</p><p>rate over the course of training, according to the formula:</p><p>lrate=d</p><p>−0.5</p><p>model</p><p>·min(step\_num</p><p>−0.5</p><p>,step\_num·warmup\_steps</p><p>−1.5</p><p>)(3)</p><p>This corresponds to increasing the learning rate linearly for the firstwarmup\_stepstraining steps,</p><p>and decreasing it thereafter proportionally to the inverse square root of the step number. We used</p><p>warmup\_steps= 4000.</p><p>5.4 Regularization</p><p>We employ three types of regularization during training:</p><p>7</p><p>Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the</p><p>English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.</p><p>Model</p><p>BLEUTraining Cost (FLOPs)</p><p>EN-DEEN-FREN-DEEN-FR</p><p>ByteNet [18]23.75</p><p>Deep-Att + PosUnk [39]39.21.0·10</p><p>20</p><p>GNMT + RL [38]24.639.922.3·10</p><p>19</p><p>1.4·10</p><p>20</p><p>ConvS2S [9]25.1640.469.6·10</p><p>18</p><p>1.5·10</p><p>20</p><p>MoE [32]26.0340.562.0·10</p><p>19</p><p>1.2·10</p><p>20</p><p>Deep-Att + PosUnk Ensemble [39]40.48.0·10</p><p>20</p><p>GNMT + RL Ensemble [38]26.3041.161.8·10</p><p>20</p><p>1.1·10</p><p>21</p><p>ConvS2S Ensemble [9]26.3641.297.7·10</p><p>19</p><p>1.2·10</p><p>21</p><p>Transformer (base model)27.338.13.3·10</p><p>18</p><p>Transformer (big)28.441.82.3·10</p><p>19</p><p>Residual DropoutWe apply dropout [33] to the output of each sub-layer, before it is added to the</p><p>sub-layer input and normalized. In addition, we apply dropout to the sums of the embeddings and the</p><p>positional encodings in both the encoder and decoder stacks. For the base model, we use a rate of</p><p>P</p><p>drop</p><p>= 0.1.</p><p>Label SmoothingDuring training, we employed label smoothing of valueε</p><p>ls</p><p>= 0.1[36]. This</p><p>hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.</p><p>6 Results</p><p>6.1 Machine Translation</p><p>On the WMT 2014 English-to-German translation task, the big transformer model (Transformer (big)</p><p>in Table 2) outperforms the best previously reported models (including ensembles) by more than2.0</p><p>BLEU, establishing a new state-of-the-art BLEU score of28.4. The configuration of this model is</p><p>listed in the bottom line of Table 3. Training took3.5days on8P100 GPUs. Even our base model</p><p>surpasses all previously published models and ensembles, at a fraction of the training cost of any of</p><p>the competitive models.</p><p>On the WMT 2014 English-to-French translation task, our big model achieves a BLEU score of41.0,</p><p>outperforming all of the previously published single models, at less than1/4the training cost of the</p><p>previous state-of-the-art model. The Transformer (big) model trained for English-to-French used</p><p>dropout rateP</p><p>drop</p><p>= 0.1, instead of0.3.</p><p>For the base models, we used a single model obtained by averaging the last 5 checkpoints, which</p><p>were written at 10-minute intervals. For the big models, we averaged the last 20 checkpoints. We</p><p>used beam search with a beam size of4and length penaltyα= 0.6[38]. These hyperparameters</p><p>were chosen after experimentation on the development set. We set the maximum output length during</p><p>inference to input length +50, but terminate early when possible [38].</p><p>Table 2 summarizes our results and compares our translation quality and training costs to other model</p><p>architectures from the literature. We estimate the number of floating point operations used to train a</p><p>model by multiplying the training time, the number of GPUs used, and an estimate of the sustained</p><p>single-precision floating-point capacity of each GPU</p><p>5</p><p>.</p><p>6.2 Model Variations</p><p>To evaluate the importance of different components of the Transformer, we varied our base model</p><p>in different ways, measuring the change in performance on English-to-German translation on the</p><p>5</p><p>We used values of 2.8, 3.7, 6.0 and 9.5 TFLOPS for K80, K40, M40 and P100, respectively.</p><p>8</p><p>Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base</p><p>model. All metrics are on the English-to-German translation development set, newstest2013. Listed</p><p>perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to</p><p>per-word perplexities.</p><p>N d</p><p>model</p><p>d</p><p>ff</p><p>h d</p><p>k</p><p>d</p><p>v</p><p>P</p><p>drop</p><p>ε</p><p>ls</p><p>trainPPLBLEUparams</p><p>steps(dev)(dev)×10</p><p>6</p><p>base65122048864640.10.1100K4.9225.865</p><p>(A)</p><p>15125125.2924.9</p><p>41281285.0025.5</p><p>1632324.9125.8</p><p>3216165.0125.4</p><p>(B)</p><p>165.1625.158</p><p>325.0125.460</p><p>(C)</p><p>26.1123.736</p><p>45.1925.350</p><p>84.8825.580</p><p>25632325.7524.528</p><p>10241281284.6626.0168</p><p>10245.1225.453</p><p>40964.7526.290</p><p>(D)</p><p>0.05.7724.6</p><p>0.24.9525.5</p><p>0.04.6725.3</p><p>0.25.4725.7</p><p>(E)positional embedding instead of sinusoids4.9225.7</p><p>big610244096160.3300K4.3326.4213</p><p>development set, newstest2013. We used beam search as described in the previous section, but no</p><p>checkpoint averaging. We present these results in Table 3.</p><p>In Table 3 rows (A), we vary the number of attention heads and the attention key and value dimensions,</p><p>keeping the amount of computation constant, as described in Section 3.2.2. While single-head</p><p>attention is 0.9 BLEU worse than the best setting, quality also drops off with too many heads.</p><p>In Table 3 rows (B), we observe that reducing the attention key sized</p><p>k</p><p>hurts model quality. This</p><p>suggests that determining compatibility is not easy and that a more sophisticated compatibility</p><p>function than dot product may be beneficial. We further observe in rows (C) and (D) that, as expected,</p><p>bigger models are better, and dropout is very helpful in avoiding over-fitting. In row (E) we replace our</p><p>sinusoidal positional encoding with learned positional embeddings [9], and observe nearly identical</p><p>results to the base model.</p><p>6.3 English Constituency Parsing</p><p>To evaluate if the Transformer can generalize to other tasks we performed experiments on English</p><p>constituency parsing. This task presents specific challenges: the output is subject to strong structural</p><p>constraints and is significantly longer than the input. Furthermore, RNN sequence-to-sequence</p><p>models have not been able to attain state-of-the-art results in small-data regimes [37].</p><p>We trained a 4-layer transformer withd</p><p>model</p><p>= 1024on the Wall Street Journal (WSJ) portion of the</p><p>Penn Treebank [25], about 40K training sentences. We also trained it in a semi-supervised setting,</p><p>using the larger high-confidence and BerkleyParser corpora from with approximately 17M sentences</p><p>[37]. We used a vocabulary of 16K tokens for the WSJ only setting and a vocabulary of 32K tokens</p><p>for the semi-supervised setting.</p><p>We performed only a small number of experiments to select the dropout, both attention and residual</p><p>(section 5.4), learning rates and beam size on the Section 22 development set, all other parameters</p><p>remained unchanged from the English-to-German base translation model. During inference, we</p><p>9</p><p>Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23</p><p>of WSJ)</p><p>ParserTrainingWSJ 23 F1</p><p>Vinyals &amp; Kaiser el al. (2014) [37]WSJ only, discriminative88.3</p><p>Petrov et al. (2006) [29]WSJ only, discriminative90.4</p><p>Zhu et al. (2013) [40]WSJ only, discriminative90.4</p><p>Dyer et al. (2016) [8]WSJ only, discriminative91.7</p><p>Transformer (4 layers)WSJ only, discriminative91.3</p><p>Zhu et al. (2013) [40]semi-supervised91.3</p><p>Huang &amp; Harper (2009) [14]semi-supervised91.3</p><p>McClosky et al. (2006) [26]semi-supervised92.1</p><p>Vinyals &amp; Kaiser el al. (2014) [37]</p><p>semi-supervised92.1</p><p>Transformer (4 layers)semi-supervised92.7</p><p>Luong et al. (2015) [23]multi-task93.0</p><p>Dyer et al. (2016) [8]generative93.3</p><p>increased the maximum output length to input length +300. We used a beam size of21andα= 0.3</p><p>for both WSJ only and the semi-supervised setting.</p><p>Our results in Table 4 show that despite the lack of task-specific tuning our model performs sur-</p><p>prisingly well, yielding better results than all previously reported models with the exception of the</p><p>Recurrent Neural Network Grammar [8].</p><p>In contrast to RNN sequence-to-sequence models [37], the Transformer outperforms the Berkeley-</p><p>Parser [29] even when training only on the WSJ training set of 40K sentences.</p><p>7 Conclusion</p><p>In this work, we presented the Transformer, the first sequence transduction model based entirely on</p><p>attention, replacing the recurrent layers most commonly used in encoder-decoder architectures with</p><p>multi-headed self-attention.</p><p>For translation tasks, the Transformer can be trained significantly faster than architectures based</p><p>on recurrent or convolutional layers. On both WMT 2014 English-to-German and WMT 2014</p><p>English-to-French translation tasks, we achieve a new state of the art. In the former task our best</p><p>model outperforms even all previously reported ensembles.</p><p>We are excited about the future of attention-based models and plan to apply them to other tasks. We</p><p>plan to extend the Transformer to problems involving input and output modalities other than text and</p><p>to investigate local, restricted attention mechanisms to efficiently handle large inputs and outputs</p><p>such as images, audio and video. Making generation less sequential is another research goals of ours.</p><p>The code we used to train and evaluate our models is available athttps://github.com/</p><p>tensorflow/tensor2tensor.</p><p>AcknowledgementsWe are grateful to Nal Kalchbrenner and Stephan Gouws for their fruitful</p><p>comments, corrections and inspiration.</p><p>References</p><p>[1]Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization.arXiv preprint</p><p>arXiv:1607.06450, 2016.</p><p>[2]Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly</p><p>learning to align and translate.CoRR, abs/1409.0473, 2014.</p><p>[3]</p><p>Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive exploration of neural</p><p>machine translation architectures.CoRR, abs/1703.03906, 2017.</p><p>[4]Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine</p><p>reading.arXiv preprint arXiv:1601.06733, 2016.</p><p>10</p><p>[5]Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk,</p><p>and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical</p><p>machine translation.CoRR, abs/1406.1078, 2014.</p><p>[6]Francois Chollet. Xception: Deep learning with depthwise separable convolutions.arXiv</p><p>preprint arXiv:1610.02357, 2016.</p><p>[7]Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation</p><p>of gated recurrent neural networks on sequence modeling.CoRR, abs/1412.3555, 2014.</p><p>[8]</p><p>Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural</p><p>network grammars. InProc. of NAACL, 2016.</p><p>[9]Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolu-</p><p>tional sequence to sequence learning.arXiv preprint arXiv:1705.03122v2, 2017.</p><p>[10]Alex Graves.Generating sequences with recurrent neural networks.arXiv preprint</p><p>arXiv:1308.0850, 2013.</p><p>[11]</p><p>Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for im-</p><p>age recognition. InProceedings of the IEEE Conference on Computer Vision and Pattern</p><p>Recognition, pages 770–778, 2016.</p><p>[12]Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in</p><p>recurrent nets: the difficulty of learning long-term dependencies, 2001.</p><p>[13]Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory.Neural computation,</p><p>9(8):1735–1780, 1997.</p><p>[14]</p><p>Zhongqiang Huang and Mary Harper. Self-training PCFG grammars with latent annotations</p><p>across languages. InProceedings of the 2009 Conference on Empirical Methods in Natural</p><p>Language Processing, pages 832–841. ACL, August 2009.</p><p>[15]Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring</p><p>the limits of language modeling.arXiv preprint arXiv:1602.02410, 2016.</p><p>[16]Łukasz Kaiser and Samy Bengio. Can active memory replace attention? InAdvances in Neural</p><p>Information Processing Systems, (NIPS), 2016.</p><p>[17]Łukasz Kaiser and Ilya Sutskever. Neural GPUs learn algorithms. InInternational Conference</p><p>on Learning Representations (ICLR), 2016.</p><p>[18]Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Ko-</p><p>ray Kavukcuoglu. Neural machine translation in linear time.arXiv preprint arXiv:1610.10099v2,</p><p>2017.</p><p>[19]</p><p>Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. Structured attention networks.</p><p>InInternational Conference on Learning Representations, 2017.</p><p>[20]</p><p>Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. InICLR, 2015.</p><p>[21]Oleksii Kuchaiev and Boris Ginsburg. Factorization tricks for LSTM networks.arXiv preprint</p><p>arXiv:1703.10722, 2017.</p><p>[22]Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen</p><p>Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding.arXiv preprint</p><p>arXiv:1703.03130, 2017.</p><p>[23]</p><p>Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task</p><p>sequence to sequence learning.arXiv preprint arXiv:1511.06114, 2015.</p><p>[24]</p><p>Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-</p><p>based neural machine translation.arXiv preprint arXiv:1508.04025, 2015.</p><p>11</p><p>[25]Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated</p><p>corpus of english: The penn treebank.Computational linguistics, 19(2):313–330, 1993.</p><p>[26]</p><p>David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In</p><p>Proceedings of the Human Language Technology Conference of the NAACL, Main Conference,</p><p>pages 152–159. ACL, June 2006.</p><p>[27]Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention</p><p>model. InEmpirical Methods in Natural Language Processing, 2016.</p><p>[28]Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive</p><p>summarization.arXiv preprint arXiv:1705.04304, 2017.</p><p>[29]</p><p>Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact,</p><p>and interpretable tree annotation. InProceedings of the 21st International Conference on</p><p>Computational Linguistics and 44th Annual Meeting of the ACL, pages 433–440. ACL, July</p><p>2006.</p><p>[30]</p><p>Ofir Press and Lior Wolf. Using the output embedding to improve language models.arXiv</p><p>preprint arXiv:1608.05859, 2016.</p><p>[31]</p><p>Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words</p><p>with subword units.arXiv preprint arXiv:1508.07909, 2015.</p><p>[32]Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton,</p><p>and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts</p><p>layer.arXiv preprint arXiv:1701.06538, 2017.</p><p>[33]Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdi-</p><p>nov. Dropout: a simple way to prevent neural networks from overfitting.Journal of Machine</p><p>Learning Research, 15(1):1929–1958, 2014.</p><p>[34]Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory</p><p>networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors,</p><p>Advances in Neural Information Processing Systems 28, pages 2440–2448. Curran Associates,</p><p>Inc., 2015.</p><p>[35]Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural</p><p>networks. InAdvances in Neural Information Processing Systems, pages 3104–3112, 2014.</p><p>[36]Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna.</p><p>Rethinking the inception architecture for computer vision.CoRR, abs/1512.00567, 2015.</p><p>[37]Vinyals &amp; Kaiser, Koo, Petrov, Sutskever, and Hinton. Grammar as a foreign language. In</p><p>Advances in Neural Information Processing Systems, 2015.</p><p>[38]Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang</p><p>Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google’s neural machine</p><p>translation system: Bridging the gap between human and machine translation.arXiv preprint</p><p>arXiv:1609.08144, 2016.</p><p>[39]Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. Deep recurrent models with</p><p>fast-forward connections for neural machine translation.CoRR, abs/1606.04199, 2016.</p><p>[40]Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. Fast and accurate</p><p>shift-reduce constituent parsing. InProceedings of the 51st Annual Meeting of the ACL (Volume</p><p>1: Long Papers), pages 434–443. ACL, August 2013.</p><p>12</p><p>Attention Visualizations</p><p>Input-Input Layer5</p><p>It</p><p>is</p><p>inthis</p><p>spirit</p><p>that</p><p>a</p><p>majority</p><p>ofAmerican</p><p>governments</p><p>have</p><p>passed</p><p>newlaws</p><p>since</p><p>2009</p><p>making</p><p>the</p><p>registration</p><p>or</p><p>voting</p><p>process</p><p>more</p><p>dif</p><p>ficult</p><p>.</p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p>It</p><p>is</p><p>in</p><p>this</p><p>spirit</p><p>that</p><p>a</p><p>majority</p><p>of</p><p>American</p><p>governments</p><p>have</p><p>passed</p><p>new</p><p>laws</p><p>since</p><p>2009</p><p>making</p><p>the</p><p>registration</p><p>or</p><p>voting</p><p>process</p><p>more</p><p>dif</p><p>ficult</p><p>.</p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p><br></p><p>Figure 3: An example of the attention mechanism following long-distance dependencies in the</p><p>encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of</p><p>the verb ‘making’, completing the phrase ‘making...more difficult’. Attentions here shown only for</p><p>the word ‘making’. Different colors represent different heads. Best viewed in color.</p><p>13</p><p>Input-Input Layer5</p><p>The</p><p>Law</p><p>willnever</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>itsapplication</p><p>should</p><p>be</p><p>just</p><p>-this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>The</p><p>Law</p><p>will</p><p>never</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>its</p><p>application</p><p>should</p><p>be</p><p>just</p><p>-</p><p>this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>Input-Input Layer5</p><p>The</p><p>Law</p><p>willnever</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>itsapplication</p><p>should</p><p>be</p><p>just</p><p>-this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>The</p><p>Law</p><p>will</p><p>never</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>its</p><p>application</p><p>should</p><p>be</p><p>just</p><p>-</p><p>this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>Figure 4: Two attention heads, also in layer 5 of 6, apparently involved in anaphora resolution. Top:</p><p>Full attentions for head 5. Bottom: Isolated attentions from just the word ‘its’ for attention heads 5</p><p>and 6. Note that the attentions are very sharp for this word.</p><p>14</p><p>Input-Input Layer5</p><p>The</p><p>Law</p><p>willnever</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>itsapplication</p><p>should</p><p>be</p><p>just</p><p>-this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>The</p><p>Law</p><p>will</p><p>never</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>its</p><p>application</p><p>should</p><p>be</p><p>just</p><p>-</p><p>this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>Input-Input Layer5</p><p>The</p><p>Law</p><p>willnever</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>itsapplication</p><p>should</p><p>be</p><p>just</p><p>-this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>The</p><p>Law</p><p>will</p><p>never</p><p>be</p><p>perfect</p><p>,</p><p>but</p><p>its</p><p>application</p><p>should</p><p>be</p><p>just</p><p>-</p><p>this</p><p>is</p><p>what</p><p>we</p><p>are</p><p>missing</p><p>,</p><p>in</p><p>my</p><p>opinion</p><p>.</p><p><br></p><p><br></p><p>Figure 5: Many of the attention heads exhibit behaviour that seems related to the structure of the</p><p>sentence. We give two such examples above, from two different heads from the encoder self-attention</p><p>at layer 5 of 6. The heads clearly learned to perform different tasks.</p><p>15</p>