Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

Jakob Uszkoreit[*]
Google Research
usz@google.com

Llion Jones[*]
Google Research
llion@google.com

Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

[*]Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

[†]Work performed while at Google Brain.

[‡]Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

1 Introduction

Recurrent neural networks, long short-term memory [13] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [35, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [38, 24, 15].

Recurrent models typically factor computation along the symbol positions of the input and output sequences. Aligning the positions to steps in computation time, they generate a sequence of hidden states h_t, as a function of the previous hidden state h_{t-1} and the input for position t. This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths, as memory constraints limit batching across examples. Recent work has achieved significant improvements in computational efficiency through factorization tricks [21] and conditional computation [32], while also improving model performance in case of the latter. The fundamental constraint of sequential computation, however, remains.

Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in the input or output sequences [2, 19]. In all but a few cases [27], however, such attention mechanisms are used in conjunction with a recurrent network.

In this work we propose the Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The Transformer allows for significantly more parallelization and can reach a new state of the art in translation quality after being trained for as little as twelve hours on eight P100 GPUs.

2 Background

The goal of reducing sequential computation also forms the foundation of the Extended Neural GPU [16], ByteNet [18] and ConvS2S [9], all of which use convolutional neural networks as basic building block, computing hidden representations in parallel for all input and output positions. In these models, the number of operations required to relate signals from two arbitrary input or output positions grows in the distance between positions, linearly for ConvS2S and logarithmically for ByteNet. This makes it more difficult to learn dependencies between distant positions [12]. In the Transformer this is reduced to a constant number of operations, albeit at the cost of reduced effective resolution due to averaging attention-weighted positions, an effect we counteract with Multi-Head Attention as described in section 3.2.

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. Self-attention has been used successfully in a variety of tasks including reading comprehension, abstractive summarization, textual entailment and learning task-independent sentence representations [4, 27, 28, 22].

End-to-end memory networks are based on a recurrent attention mechanism instead of sequence-aligned recurrence and have been shown to perform well on simple-language question answering and language modeling tasks [34].

To the best of our knowledge, however, the Transformer is the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. In the following sections, we will describe the Transformer, motivate self-attention and discuss its advantages over models such as [17, 18] and [9].

3 Model Architecture

Most competitive neural sequence transduction models have an encoder-decoder structure [5, 2, 35]. Here, the encoder maps an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $\mathbf{z} = (z_1, ..., z_n)$. Given $\mathbf{z}$, the decoder then generates an output sequence $(y_1, ..., y_m)$ of symbols one element at a time. At each step the model is auto-regressive [10], consuming the previously generated symbols as additional input when generating the next.

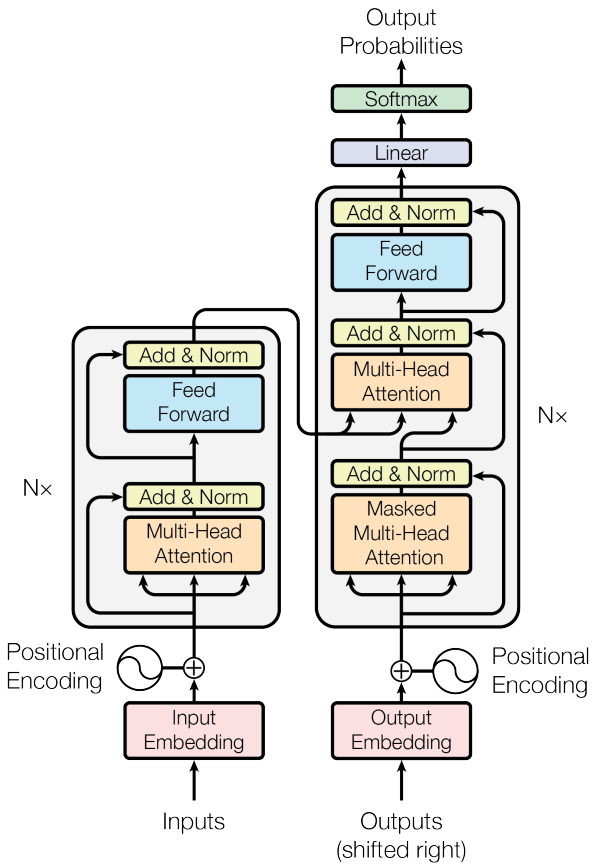


Figure 1: The Transformer - model architecture.

The Transformer follows this overall architecture using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, shown in the left and right halves of Figure 1, respectively.

3.1 Encoder and Decoder Stacks

Encoder: The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second is a simple, position-wise fully connected feed-forward network. We employ a residual connection [11] around each of the two sub-layers, followed by layer normalization [1]. That is, the output of each sub-layer is $\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$, where $\mathrm{Sublayer}(x)$ is the function implemented by the sub-layer itself. To facilitate these residual connections, all sub-layers in the model, as well as the embedding layers, produce outputs of dimension $d_{\text{model}} = 512$.

Decoder: The decoder is also composed of a stack of $N = 6$ identical layers. In addition to the two sub-layers in each encoder layer, the decoder inserts a third sub-layer, which performs multi-head attention over the output of the encoder stack. Similar to the encoder, we employ residual connections around each of the sub-layers, followed by layer normalization. We also modify the self-attention sub-layer in the decoder stack to prevent positions from attending to subsequent positions. This masking, combined with fact that the output embeddings are offset by one position, ensures that the predictions for position i can depend only on the known outputs at positions less than i.

3.2 Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum

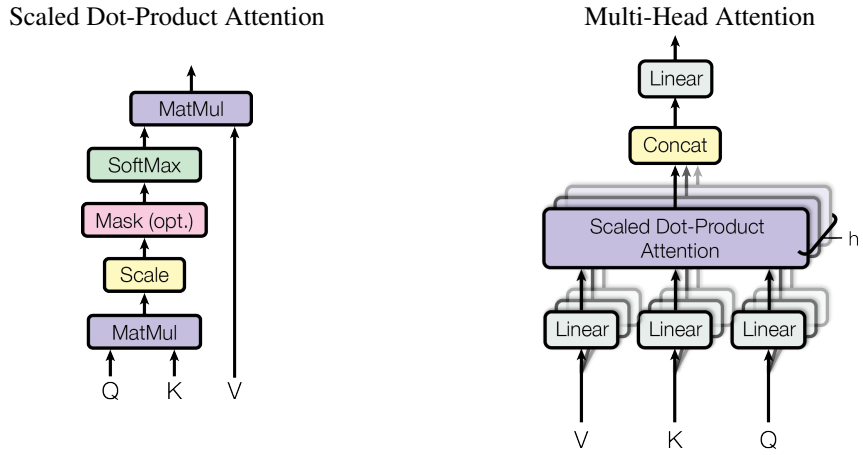


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

3.2.1 Scaled Dot-Product Attention

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

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

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

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

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

3.2.2 Multi-Head Attention

Instead of performing a single attention function with d_{model}-dimensional keys, values and queries, we found it beneficial to linearly project the queries, keys and values h times with different, learned linear projections to d_k, d_k and d_v dimensions, respectively. On each of these projected versions of queries, keys and values we then perform the attention function in parallel, yielding d_v-dimensional

[4] To illustrate why the dot products get large, assume that the components of q and k are independent random variables with mean 0 and variance 1. Then their dot product, $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$, has mean 0 and variance d_k.

output values. These are concatenated and once again projected, resulting in the final values, as depicted in Figure 2.

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this.

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O \\ \text{where } \text{head}_\text{i} &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)\end{aligned}$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_\text{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_\text{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_\text{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_\text{model}}$.

In this work we employ $h = 8$ parallel attention layers, or heads. For each of these we use $d_k = d_v = d_\text{model}/h = 64$. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality.

3.2.3 Applications of Attention in our Model

The Transformer uses multi-head attention in three different ways:

- In "encoder-decoder attention" layers, the queries come from the previous decoder layer, and the memory keys and values come from the output of the encoder. This allows every position in the decoder to attend over all positions in the input sequence. This mimics the typical encoder-decoder attention mechanisms in sequence-to-sequence models such as [38, 2, 9].
- The encoder contains self-attention layers. In a self-attention layer all of the keys, values and queries come from the same place, in this case, the output of the previous layer in the encoder. Each position in the encoder can attend to all positions in the previous layer of the encoder.
- Similarly, self-attention layers in the decoder allow each position in the decoder to attend to all positions in the decoder up to and including that position. We need to prevent leftward information flow in the decoder to preserve the auto-regressive property. We implement this inside of scaled dot-product attention by masking out (setting to $-\infty$) all values in the input of the softmax which correspond to illegal connections. See Figure 2.

3.3 Position-wise Feed-Forward Networks

In addition to attention sub-layers, each of the layers in our encoder and decoder contains a fully connected feed-forward network, which is applied to each position separately and identically. This consists of two linear transformations with a ReLU activation in between.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{2}$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. Another way of describing this is as two convolutions with kernel size 1. The dimensionality of input and output is $d_\text{model} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$.

3.4 Embeddings and Softmax

Similarly to other sequence transduction models, we use learned embeddings to convert the input tokens and output tokens to vectors of dimension d_model. We also use the usual learned linear transformation and softmax function to convert the decoder output to predicted next-token probabilities. In our model, we share the same weight matrix between the two embedding layers and the pre-softmax linear transformation, similar to [30]. In the embedding layers, we multiply those weights by $\sqrt{d_\text{model}}$.

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

3.5 Positional Encoding

Since our model contains no recurrence and no convolution, in order for the model to make use of the order of the sequence, we must inject some information about the relative or absolute position of the tokens in the sequence. To this end, we add "positional encodings" to the input embeddings at the bottoms of the encoder and decoder stacks. The positional encodings have the same dimension d_{model} as the embeddings, so that the two can be summed. There are many choices of positional encodings, learned and fixed [9].

In this work, we use sine and cosine functions of different frequencies:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. We chose this function because we hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.

We also experimented with using learned positional embeddings [9] instead, and found that the two versions produced nearly identical results (see Table 3 row (E)). We chose the sinusoidal version because it may allow the model to extrapolate to sequence lengths longer than the ones encountered during training.

4 Why Self-Attention

In this section we compare various aspects of self-attention layers to the recurrent and convolutional layers commonly used for mapping one variable-length sequence of symbol representations $(x_1, ..., x_n)$ to another sequence of equal length $(z_1, ..., z_n)$, with $x_i, z_i \in \mathbb{R}^d$, such as a hidden layer in a typical sequence transduction encoder or decoder. Motivating our use of self-attention we consider three desiderata.

One is the total computational complexity per layer. Another is the amount of computation that can be parallelized, as measured by the minimum number of sequential operations required.

The third is the path length between long-range dependencies in the network. Learning long-range dependencies is a key challenge in many sequence transduction tasks. One key factor affecting the ability to learn such dependencies is the length of the paths forward and backward signals have to traverse in the network. The shorter these paths between any combination of positions in the input and output sequences, the easier it is to learn long-range dependencies [12]. Hence we also compare the maximum path length between any two input and output positions in networks composed of the different layer types.

As noted in Table 1, a self-attention layer connects all positions with a constant number of sequentially executed operations, whereas a recurrent layer requires $O(n)$ sequential operations. In terms of computational complexity, self-attention layers are faster than recurrent layers when the sequence

length n is smaller than the representation dimensionality d, which is most often the case with sentence representations used by state-of-the-art models in machine translations, such as word-piece [38] and byte-pair [31] representations. To improve computational performance for tasks involving very long sequences, self-attention could be restricted to considering only a neighborhood of size r in the input sequence centered around the respective output position. This would increase the maximum path length to $O(n/r)$. We plan to investigate this approach further in future work.

A single convolutional layer with kernel width $k < n$ does not connect all pairs of input and output positions. Doing so requires a stack of $O(n/k)$ convolutional layers in the case of contiguous kernels, or $O(log_k(n))$ in the case of dilated convolutions [18], increasing the length of the longest paths between any two positions in the network. Convolutional layers are generally more expensive than recurrent layers, by a factor of k. Separable convolutions [6], however, decrease the complexity considerably, to $O(k \cdot n \cdot d + n \cdot d^2)$. Even with $k = n$, however, the complexity of a separable convolution is equal to the combination of a self-attention layer and a point-wise feed-forward layer, the approach we take in our model.

As side benefit, self-attention could yield more interpretable models. We inspect attention distributions from our models and present and discuss examples in the appendix. Not only do individual attention heads clearly learn to perform different tasks, many appear to exhibit behavior related to the syntactic and semantic structure of the sentences.

5 Training

This section describes the training regime for our models.

5.1 Training Data and Batching

We trained on the standard WMT 2014 English-German dataset consisting of about 4.5 million sentence pairs. Sentences were encoded using byte-pair encoding [3], which has a shared source-target vocabulary of about 37000 tokens. For English-French, we used the significantly larger WMT 2014 English-French dataset consisting of 36M sentences and split tokens into a 32000 word-piece vocabulary [38]. Sentence pairs were batched together by approximate sequence length. Each training batch contained a set of sentence pairs containing approximately 25000 source tokens and 25000 target tokens.

5.2 Hardware and Schedule

We trained our models on one machine with 8 NVIDIA P100 GPUs. For our base models using the hyperparameters described throughout the paper, each training step took about 0.4 seconds. We trained the base models for a total of 100,000 steps or 12 hours. For our big models,(described on the bottom line of table 3), step time was 1.0 seconds. The big models were trained for 300,000 steps (3.5 days).

5.3 Optimizer

We used the Adam optimizer [20] with $\beta_1 = 0.9$, $\beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We varied the learning rate over the course of training, according to the formula:

$$lrate = d_{\text{model}}^{-0.5} \cdot \min(step\_num^{-0.5}, step\_num \cdot warmup\_steps^{-1.5}) \tag{3}$$

This corresponds to increasing the learning rate linearly for the first $warmup\_steps$ training steps, and decreasing it thereafter proportionally to the inverse square root of the step number. We used $warmup\_steps = 4000$.

5.4 Regularization

We employ three types of regularization during training:

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

Residual Dropout We apply dropout [33] to the output of each sub-layer, before it is added to the sub-layer input and normalized. In addition, we apply dropout to the sums of the embeddings and the positional encodings in both the encoder and decoder stacks. For the base model, we use a rate of $P_{drop} = 0.1$.

Label Smoothing During training, we employed label smoothing of value $\epsilon_{ls} = 0.1$ [36]. This hurts perplexity, as the model learns to be more unsure, but improves accuracy and BLEU score.

6 Results

6.1 Machine Translation

On the WMT 2014 English-to-German translation task, the big transformer model (Transformer (big) in Table 2) outperforms the best previously reported models (including ensembles) by more than 2.0 BLEU, establishing a new state-of-the-art BLEU score of 28.4. The configuration of this model is listed in the bottom line of Table 3. Training took 3.5 days on 8 P100 GPUs. Even our base model surpasses all previously published models and ensembles, at a fraction of the training cost of any of the competitive models.

On the WMT 2014 English-to-French translation task, our big model achieves a BLEU score of 41.0, outperforming all of the previously published single models, at less than $1/4$ the training cost of the previous state-of-the-art model. The Transformer (big) model trained for English-to-French used dropout rate $P_{drop} = 0.1$, instead of 0.3.

For the base models, we used a single model obtained by averaging the last 5 checkpoints, which were written at 10-minute intervals. For the big models, we averaged the last 20 checkpoints. We used beam search with a beam size of 4 and length penalty $\alpha = 0.6$ [38]. These hyperparameters were chosen after experimentation on the development set. We set the maximum output length during inference to input length + 50, but terminate early when possible [38].

Table 2 summarizes our results and compares our translation quality and training costs to other model architectures from the literature. We estimate the number of floating point operations used to train a model by multiplying the training time, the number of GPUs used, and an estimate of the sustained single-precision floating-point capacity of each GPU [5].

6.2 Model Variations

To evaluate the importance of different components of the Transformer, we varied our base model in different ways, measuring the change in performance on English-to-German translation on the

[5]We used values of 2.8, 3.7, 6.0 and 9.5 TFLOPS for K80, K40, M40 and P100, respectively.

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

development set, newstest2013. We used beam search as described in the previous section, but no checkpoint averaging. We present these results in Table 3.

In Table 3 rows (A), we vary the number of attention heads and the attention key and value dimensions, keeping the amount of computation constant, as described in Section 3.2.2. While single-head attention is 0.9 BLEU worse than the best setting, quality also drops off with too many heads.

In Table 3 rows (B), we observe that reducing the attention key size d_k hurts model quality. This suggests that determining compatibility is not easy and that a more sophisticated compatibility function than dot product may be beneficial. We further observe in rows (C) and (D) that, as expected, bigger models are better, and dropout is very helpful in avoiding over-fitting. In row (E) we replace our sinusoidal positional encoding with learned positional embeddings [9], and observe nearly identical results to the base model.

6.3 English Constituency Parsing

To evaluate if the Transformer can generalize to other tasks we performed experiments on English constituency parsing. This task presents specific challenges: the output is subject to strong structural constraints and is significantly longer than the input. Furthermore, RNN sequence-to-sequence models have not been able to attain state-of-the-art results in small-data regimes [37].

We trained a 4-layer transformer with $d_{model} = 1024$ on the Wall Street Journal (WSJ) portion of the Penn Treebank [25], about 40K training sentences. We also trained it in a semi-supervised setting, using the larger high-confidence and BerkleyParser corpora from with approximately 17M sentences [37]. We used a vocabulary of 16K tokens for the WSJ only setting and a vocabulary of 32K tokens for the semi-supervised setting.

We performed only a small number of experiments to select the dropout, both attention and residual (section 5.4), learning rates and beam size on the Section 22 development set, all other parameters remained unchanged from the English-to-German base translation model. During inference, we

Table 4: The Transformer generalizes well to English constituency parsing (Results are on Section 23 of WSJ)

Parser	Training	WSJ 23 F1
Vinyals & Kaiser el al. (2014) [37]	WSJ only, discriminative	88.3
Petrov et al. (2006) [29]	WSJ only, discriminative	90.4
Zhu et al. (2013) [40]	WSJ only, discriminative	90.4
Dyer et al. (2016) [8]	WSJ only, discriminative	91.7
Transformer (4 layers)	WSJ only, discriminative	91.3
Zhu et al. (2013) [40]	semi-supervised	91.3
Huang & Harper (2009) [14]	semi-supervised	91.3
McClosky et al. (2006) [26]	semi-supervised	92.1
Vinyals & Kaiser el al. (2014) [37]	semi-supervised	92.1
Transformer (4 layers)	semi-supervised	92.7
Luong et al. (2015) [23]	multi-task	93.0
Dyer et al. (2016) [8]	generative	93.3

increased the maximum output length to input length + 300. We used a beam size of 21 and $\alpha = 0.3$ for both WSJ only and the semi-supervised setting.

Our results in Table 4 show that despite the lack of task-specific tuning our model performs surprisingly well, yielding better results than all previously reported models with the exception of the Recurrent Neural Network Grammar [8].

In contrast to RNN sequence-to-sequence models [37], the Transformer outperforms the Berkeley-Parser [29] even when training only on the WSJ training set of 40K sentences.

7 Conclusion

In this work, we presented the Transformer, the first sequence transduction model based entirely on attention, replacing the recurrent layers most commonly used in encoder-decoder architectures with multi-headed self-attention.

For translation tasks, the Transformer can be trained significantly faster than architectures based on recurrent or convolutional layers. On both WMT 2014 English-to-German and WMT 2014 English-to-French translation tasks, we achieve a new state of the art. In the former task our best model outperforms even all previously reported ensembles.

We are excited about the future of attention-based models and plan to apply them to other tasks. We plan to extend the Transformer to problems involving input and output modalities other than text and to investigate local, restricted attention mechanisms to efficiently handle large inputs and outputs such as images, audio and video. Making generation less sequential is another research goals of ours.

The code we used to train and evaluate our models is available at `https://github.com/tensorflow/tensor2tensor`.

Acknowledgements We are grateful to Nal Kalchbrenner and Stephan Gouws for their fruitful comments, corrections and inspiration.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[3] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906, 2017.

[4] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[5] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.

[6] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint arXiv:1610.02357*, 2016.

[7] Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[8] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural network grammars. In *Proc. of NAACL*, 2016.

[9] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122v2*, 2017.

[10] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[12] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[14] Zhongqiang Huang and Mary Harper. Self-training PCFG grammars with latent annotations across languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 832–841. ACL, August 2009.

[15] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*, 2016.

[16] Łukasz Kaiser and Samy Bengio. Can active memory replace attention? In *Advances in Neural Information Processing Systems, (NIPS)*, 2016.

[17] Łukasz Kaiser and Ilya Sutskever. Neural GPUs learn algorithms. In *International Conference on Learning Representations (ICLR)*, 2016.

[18] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099v2*, 2017.

[19] Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. Structured attention networks. In *International Conference on Learning Representations*, 2017.

[20] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[21] Oleksii Kuchaiev and Boris Ginsburg. Factorization tricks for LSTM networks. *arXiv preprint arXiv:1703.10722*, 2017.

[22] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[23] Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*, 2015.

[24] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[25] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[26] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159. ACL, June 2006.

[27] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model. In *Empirical Methods in Natural Language Processing*, 2016.

[28] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[29] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 433–440. ACL, July 2006.

[30] Ofir Press and Lior Wolf. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*, 2016.

[31] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[32] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[33] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[34] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc., 2015.

[35] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[36] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[37] Vinyals & Kaiser, Koo, Petrov, Sutskever, and Hinton. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, 2015.

[38] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[39] Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. Deep recurrent models with fast-forward connections for neural machine translation. *CoRR*, abs/1606.04199, 2016.

[40] Muhua Zhu, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. Fast and accurate shift-reduce constituent parsing. In *Proceedings of the 51st Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 434–443. ACL, August 2013.

Attention Visualizations

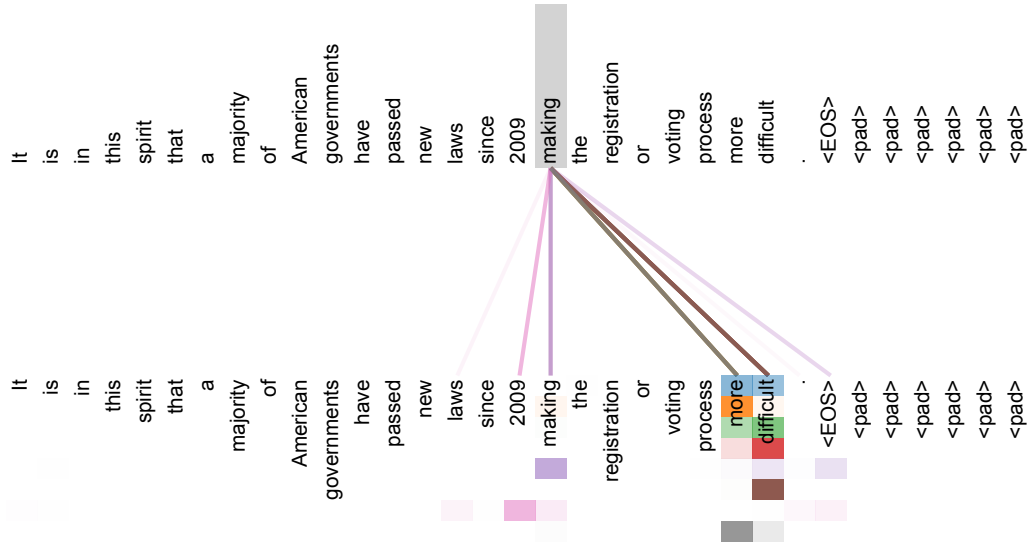


Figure 3: An example of the attention mechanism following long-distance dependencies in the encoder self-attention in layer 5 of 6. Many of the attention heads attend to a distant dependency of the verb 'making', completing the phrase 'making...more difficult'. Attentions here shown only for the word 'making'. Different colors represent different heads. Best viewed in color.

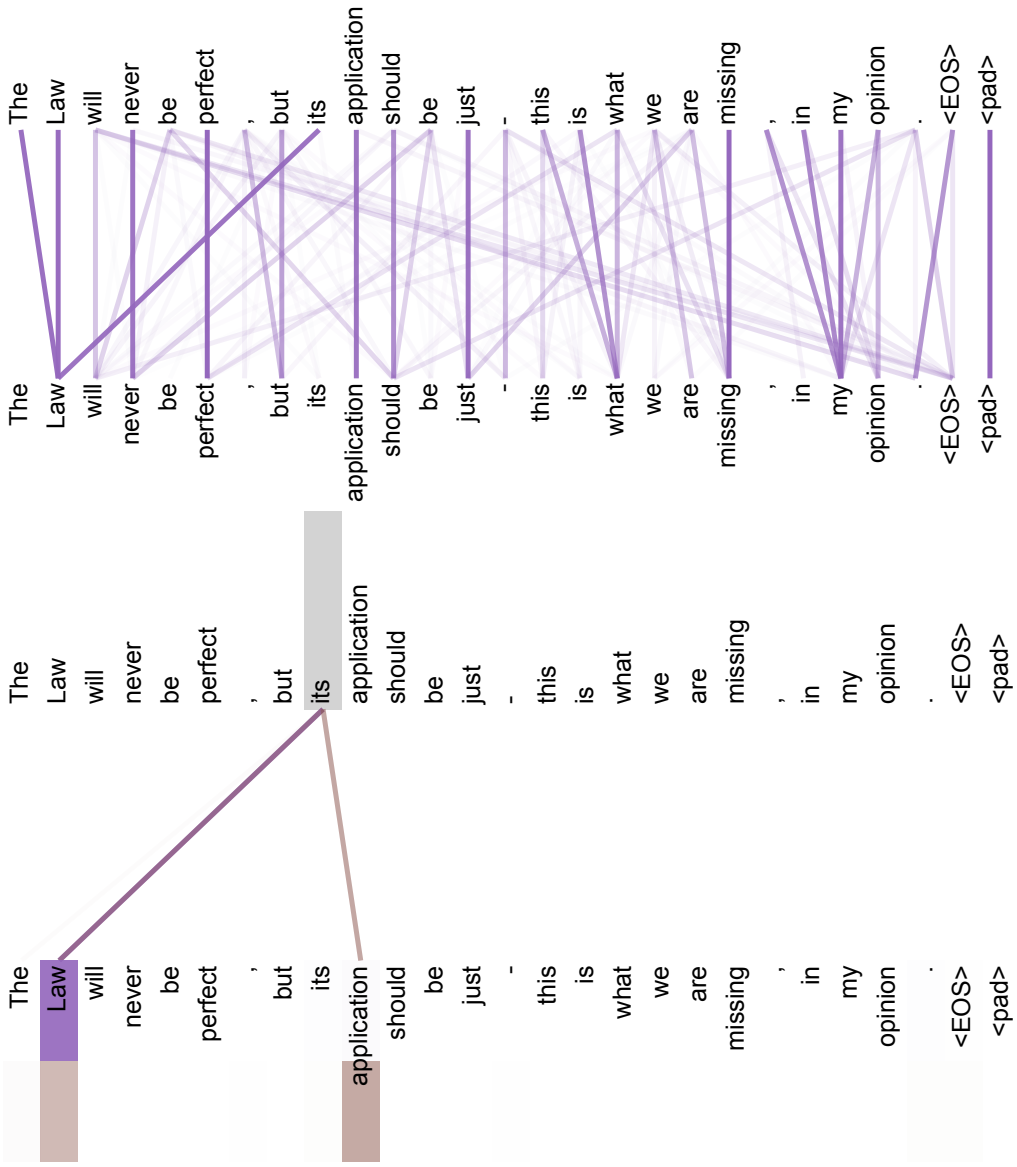


Figure 4: Two attention heads, also in layer 5 of 6, apparently involved in anaphora resolution. Top: Full attentions for head 5. Bottom: Isolated attentions from just the word 'its' for attention heads 5 and 6. Note that the attentions are very sharp for this word.

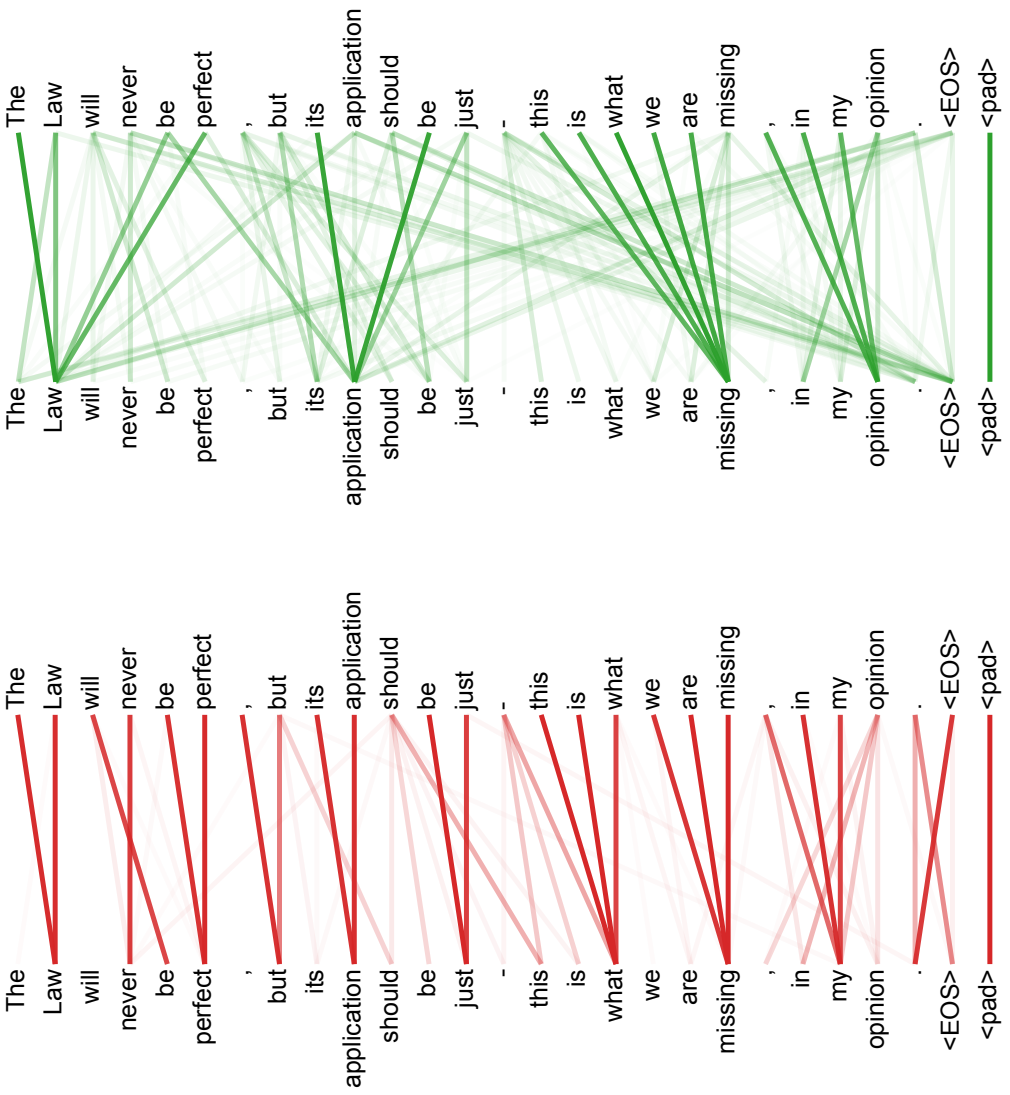


Figure 5: Many of the attention heads exhibit behaviour that seems related to the structure of the sentence. We give two such examples above, from two different heads from the encoder self-attention at layer 5 of 6. The heads clearly learned to perform different tasks.

资料分析考试大纲：资料分析主要测查报考者对文字、数字、图表等统计性资料的综合理解与分析加工能力。

难点：找数、列式、计算

第一章：八大常用速算技巧

一、4 个常用速算小技巧

1、一个数×1.5，等于这个数+本身的一半

120×1.5=

124×1.5=

2、

（1）一个数×1.1，等于这个数错位相加

120×1.1=

124×1.1=

（2）一个数×0.9，等于这个数错位相减

120×0.9=

124×0.9=

【例 1】 36785.2×（1+10.2%）=

A. 38458

B. 40537

C. 45614

D. 54324

【例 2】 36785.2×（1-10.2%）=

A. 31256

B. 33033

C. 35142

D. 38476

3、

一个数÷5，等于这个数×2，小数点移一位。

一个数÷25，等于这个数×4，小数点移两位。

一个数÷125，等于这个数×8，小数点移三位。

$\frac{24}{5}$ = _____ $\frac{36}{5}$ =_____

$\frac{24}{25}$ = _____ $\frac{36}{25}$ =_____

$\frac{24}{125}$ = _____ $\frac{36}{125}$ =_____

4、常用分数（必须会背）

高照教你百化分

①不用背，我也会

50%=_____、33.3%=_____、25%=_____、20%=_____、10%=_____

②记住（$\frac{1}{8}$~$\frac{1}{13}$），加和为 20 （整数部分+分母）

12.5%=_____、11.1%=_____、9.1%=_____、8.3%=_____、7.7%=_____

③记住（16、6）和（14、7）互换的两对

16.7%=_____、6.25%=_____、14.3%=_____、7.1%=_____

④记住（17、18、19），5.963

5.9%=_____、5.6%=_____、5.3%=_____

⑤就记住 6.7%=$\frac{1}{15}$

【例 3】$\frac{23.2}{24.8\%}$=

A.60.3

B.77.8

C.84.1

D.93.5

【例 4】$\frac{65391}{12.6\%}$=

A.59.3 万

B.56.7 万

C.54.4 万

D.51.9 万

【例 5】$\frac{262}{66.7\%}$=

A.375

B.383

C.393

D.408

【例 6】**63714×14.3%=**

A.11331

B.10203

C.9111

D.8425

2022 年，...直播电商行业交易额为 3.5 万亿元，直播电商渗透率（直播电商行业渗透率=直播电商行业交易额/网络零售交易额）为 25.3%。

【例 7】（2024 江苏）2022 年我国网络零售交易额为（　）（注：网友回忆版没有 AB 选项）

C.13.8 万亿

D.15.3 万亿

答案：1-5:BBDDC　6-7:CC

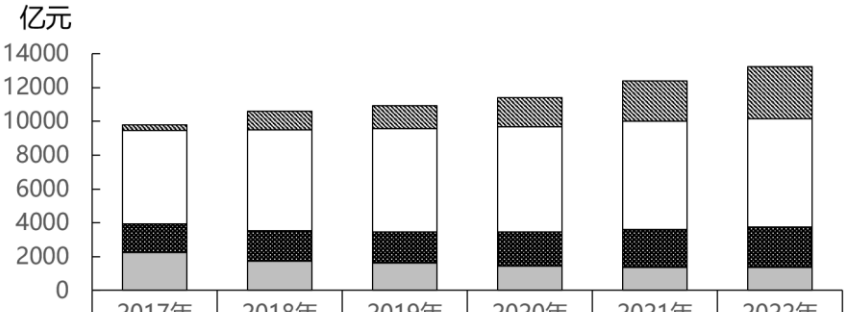
二、加法:高位叠加

两位数加减：口算

A	B	A+B	A-B
69	21		
95	32		
47	32		
76	17		
61	48		
93	45		

三位数加法：高位叠加、不列式、少动笔

A	**B**	**A+B**
425	**332**	
646	**758**	
293	**476**	
314	**253**	
353	**677**	



	2017年	2018年	2019年	2020年	2021年	2022年
新兴业务	309	1084	1374	1737	2372	3072
移动数据流量业务	5528	5984	6096	6204	6381	6397
互联网宽带接入业务	1674	1784	1855	2027	2243	2402
语音业务	2263	1744	1613	1435	1375	1364

图1 2017~2022年全国主要电信业务收入

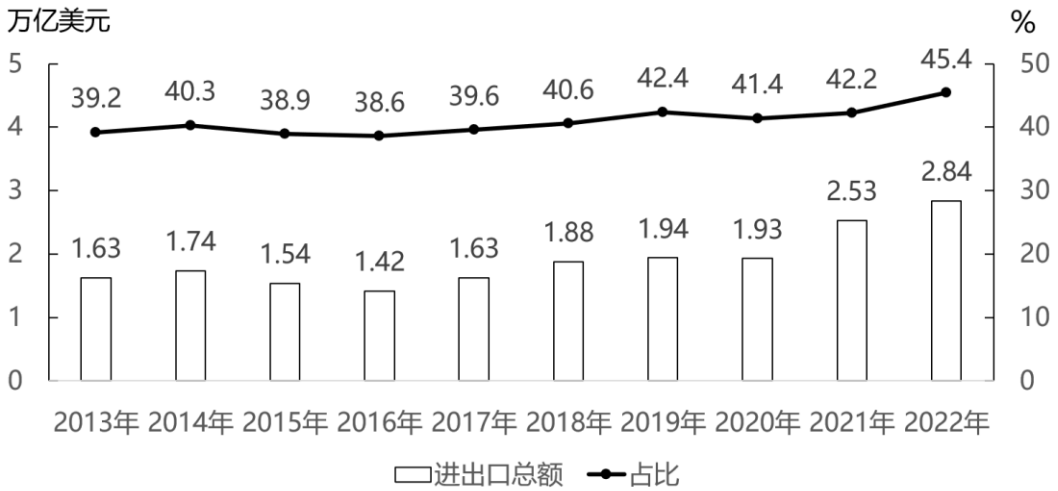
【例 1】（2024 国考）2017~2022 年，全国移动数据流量业务收入总额在以下哪个范围内？

A.不到 3.4 万亿元

B.超过 3.6 万亿元

C.3.4 万亿—3.5 万亿元之间

D.3.5 万亿—3.6 万亿元之间



2013~2022年中国与共建国家进出口总额及其占中国外贸总值比重

【例 2】（2024 联考）2013-2022 年，中国与共建国家进出口总额合计约为：

A.至少 20 万亿美元

B.在 17-20 万亿美元之间

C.在 14-17 万亿美元之间

D.至多 14 万亿美元

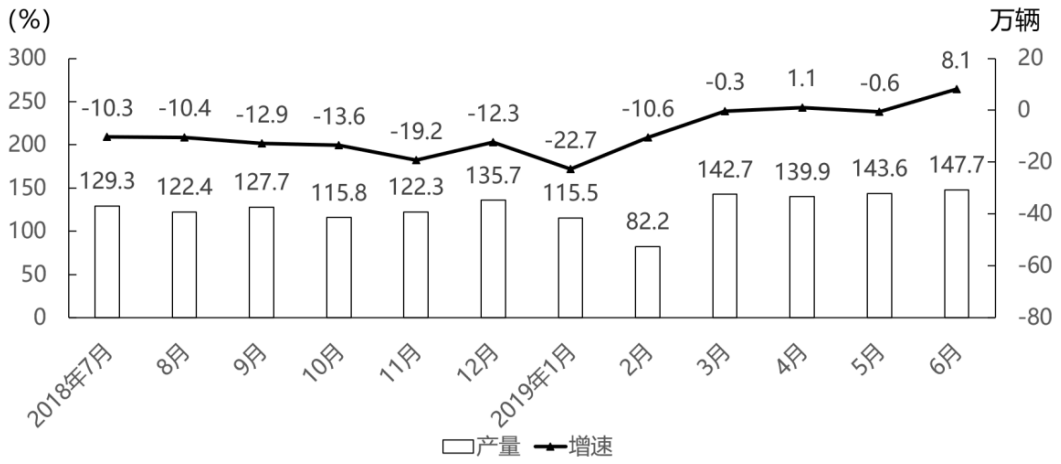


图1 2018年7月~2019年6月我国摩托车产量及同比增速

【例 3】（2024 四川）将①2018 年第三季度②2018 年第四季度③2019 年第一季度④2019 年第二季度，我国摩托车产量从低到高排列，下列排序正确的是：

A.③<②<①<④

B.③<①<②<④

C.③<④<②<①

D.③<①<④<②

答案：1-3: BBA

三、减法:划线减法

（一）临界值：（插入临界值）

1、714-688=

2、613-587=

3、423-389=

（二）普通数值（划线方法，以好算、少借位为前提）

A	B	A-B
816	634	
974	546	
697	516	
890	362	
344	282	
756	347	

2022 年，京津冀地区生产总值合计 10.0 万亿元，是 2013 年的 1.8 倍。其中，北京、河北跨越 4 万亿元量级，均为 4.2 万亿元，分别是 2013 年的 2.0 倍和 1.7 倍；天津 1.6 万亿元，是 2013 年的 1.6 倍。京津冀第一产业、第二产业、第三产业增加值占生产总值比重构成由 2013 年的 6.2：35.7：58.1 变化为 2022 年的 4.8：29.6：65.6。京津冀三地第三产业增加值占生产总值比重分别为 83.8%、61.3％和 49.4%，较 2013 年分别提高 4.3、7.2 和 8.4 个百分点。

【例 1】（2024 国考）2013 年，北京第三产业增加值占其生产总值比重比天津高多少个百分点？

A.25.4

B.22.5

C.19.6

D.16.7

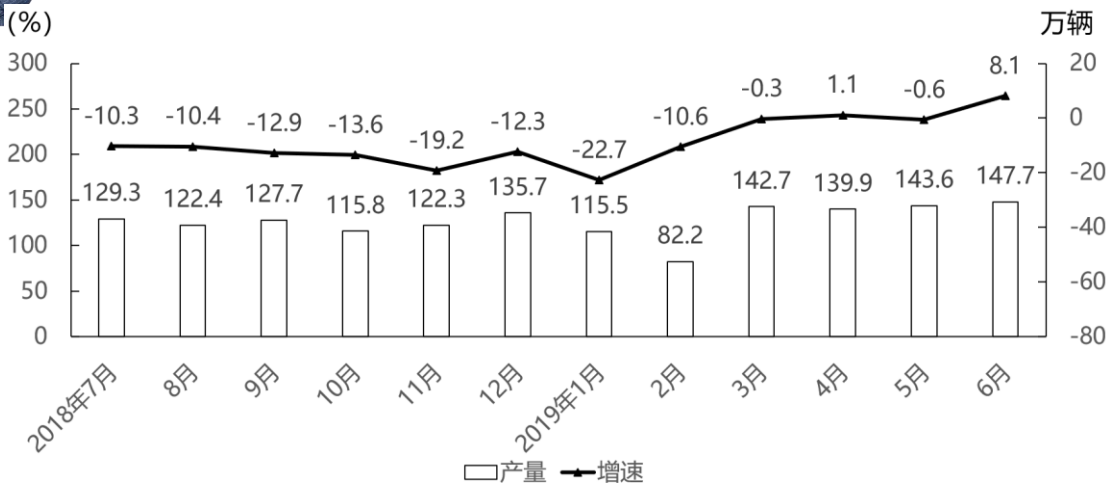


图1 2018年7月~2019年6月我国摩托车产量及同比增速

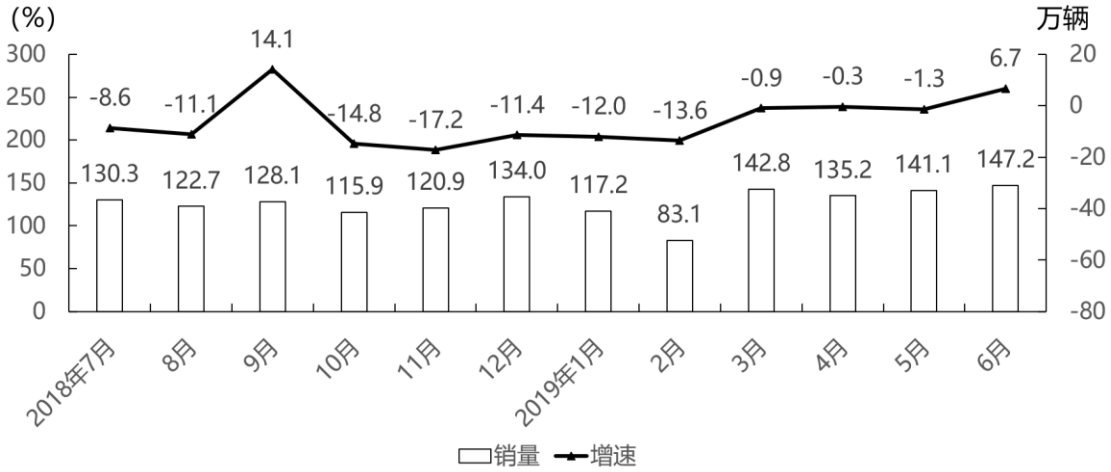


图2 2018年7月~2019年6月我国摩托车销量及同比增速

【例 2】（2024 四川）2018 年 7 月~2019 年 6 月，我国摩托车产量与销量相差不超过 0.5 万辆的月份有多少个?

A.4

B.5

C.6

D.7

2022~2023年上半年某市经济社会发展主要指标

类型	2022年				2023年	
	1~3月累计	1~6月累计	1~9月累计	1~12月累计	1~3月累计	1~6月累计
地区生产总值（亿元）	4004	7879	12221	16908	4230	8317
第三产业增加值占比（%）	64.3	63.3	62.9	62.2	65.4	64.8
工业用电量（亿千瓦时）	81.6	169.8	264.3	355.3	81.3	174.3
固定资产投资（亿元）	1323	3040	4512	5875	1406	3111
社会消费品零售总额（亿元）	2112	4000	5833	7832	2252	4358
进出口总额（亿元）	1388	2988	4556	6292	1443	2915
出口占进出口总额的比重（%）	58.1	59.9	59.7	60.8	57.8	58.1
实际利用外资（亿美元）	22.8	35.3	41.3	48.5	33.0	39.6

【例 3】（2024 江苏）2023 年二季度，该市工业用电量同比增加（　）（注：网友回忆版没有 CD 选项）

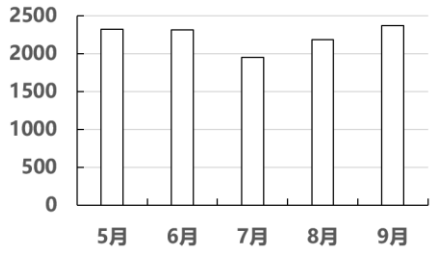
A.4.5 亿千瓦时

B.4.8 亿千瓦时

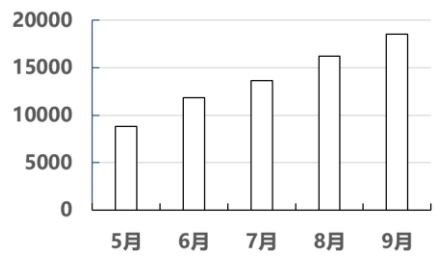
2021年4月~2021年9月软件业务各类月末收入累计值（亿元）

行业 / 月份	软件业务	软件产品	信息技术服务	嵌入式系统软件
1月~4月	25719	6823	16222	2289
1月~5月	33893	9142	21314	2921
1月~6月	44198	11451	28319	3696
1月~7月	51441	13403	32744	4413
1月~8月	59710	15582	37951	5142
1月~9月	69007	17951	43980	5850

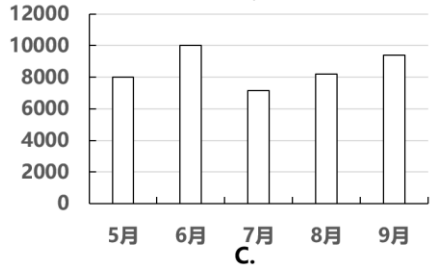
【例 4】（2023 联考）下图能正确反映 2021 年 5 月至 9 月软件产品收入（亿元）的是：



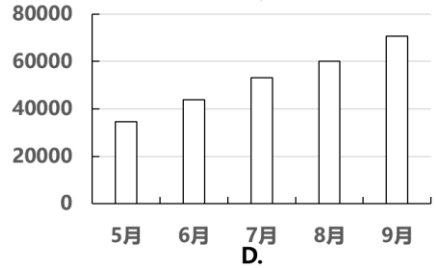
A.



B.



C.



D.

答案：1-4:ABBA

四、乘法（化乘为加、凑整、百化分）

（1）两位数×个位数：化乘为加

A	B	A×B
92	4	
84	5	
75	6	
65	7	
56	8	
46	9	

（2）两位数×两位数：化乘为加

操作方法：前前后后，里里外外

A	B	A×B
85	36	
42	43	
56	36	
37	24	
26	26	

（3）百化分：根据选项差距

估算：3224×12.6%≈3224×12.5%=3224×$\frac{1}{8}$

精算：3224×12.6%=3224×（12.5%+0.1%）

2022 年某省实现地区生产总值 87435.1 亿元，按不变价格计算，比上年增长 3.9%。分产业看，第一产业增加值 6298.6 亿元，增长 4.3%，三次产业结构为 7.2: 40.0:52.8。

【例 1】（2024 山东）2022 年，第三产业增加值比第二产业增加值大约高：

A.39870.4 亿元

B.28678.7 亿元

C.12130.7 亿元

D.11191.7 亿元

2022 年废有色金属中废铅回收重量同比增长 5.56%。

2021~2022年我国十个品种再生资源回收情况

序号	名称	回收重量（万吨）		回收金额（亿元）	
		2021年	2022年	2021年	2022年
1	废钢铁	25021.0	24081.0	7523.6	6911.2
2	废有色金属	1348.0	1375.0	2878.5	2959.7
3	废塑料	1900.0	1800.0	1050.0	1050.0
4	废纸	6491.0	6585.0	1493.0	1402.6
5	废轮胎	640.0	675.0	76.8	101.3
6	废弃电器电子产品	463.0	415.0	222.4	227.4
7	报废机动车	678.5	820.7	276.9	311.9
8	废旧纺织品	475.0	415.0	26.1	16.6
9	废玻璃	1005.0	850.0	48.0	38.3
10	废电池（铅酸电池除外）	42.0	51.0	99.7	121.6

2021年废有色金属中各类废金属回收重量占比情况

【例 2】（2024 联考）2022 年我国废有色金属中废铅回收重量约为：

A.261 万吨

B.275 万吨

C.285 万吨

D.291 万吨

2022 年，我国规模以上互联网和相关服务企业（以下简称互联网企业）完成互联网业务收入 14590 亿元，同比下降 1.1%。

分地区运行情况。2022 年，东部地区完成互联网业务收入同比下降 0.2%，占全国互联网业务收入的比重为 90.8%。中部地区完成互联网业务收入 570.8 亿元，同比增长 1.1%。西部地区完成互联网业务收入 721.1 亿元，同比下降 16.9%。东北地区完成互联网业务收入 53.6 亿元，同比增长 5.5%。

【例 3】（2024 天津事业单位）2022 年东部地区完成互联网业务收入为（　　）亿元。

A.13248

B.13325

C.13478

D.13562

答案：1-3:DCA

五、分数拆分思想：化 1 法、化半法、凑整法

【示例 1】$\frac{145}{142}=\frac{142+3}{142}=\frac{142}{142}+\frac{3}{142}\approx 1+2\%$

【示例 2】$\frac{74}{142}=\frac{71+3}{142}=\frac{71}{142}+\frac{3}{142}\approx 50\%+2\%$

【例 1】$\boldsymbol{\frac{254}{248}}$

A.100.4%

B.101.4%

C.102.4%

D.103.4%

【例 2】$\boldsymbol{\frac{248}{254}}$

A.95.6%

B.96.6%

C.97.6%

D.98.6%

【例 3】$\frac{1660.49}{3480.70}$

A.45.5%

B.46.2%

C.47.7%

D.48.7%

【例 4】$\frac{1144}{2309}$

A.48.8%

B.49.5%

C.5.12%

D.5.37%

2022 年一季度部分省市软件和信息技术服务业完成情况

	软件业务收入		其中：信息技术服务收入	
	本年累计（亿元）	同比增长（%）	本年累计（亿元）	同比增长（%）
全国	20059.67	11.6	13102.09	13.7
贵州	144.3	100.8	133.14	101.4
北京	4394.68	13.2	2894.97	13.9
天津	420.08	2.1	273.05	1.3
河北	84.18	20.3	73.43	21.7
上海	1647.5	17.6	1126.03	22.0
江苏	2805.58	10.5	1697.25	11.7
浙江	1725.91	5.4	1357.55	5.3
福建	414.63	15.7	231.61	17.5
山东	1274.51	18.0	618.44	24.8
广东	3785.08	6.1	2610.26	10.2

【例 5】（2024 浙江）2022 年一季度，表中信息技术服务收入累计值排名前三的省市，其信息技术服务收入累计值之和约占全国累计值的多少？

A.51%

B.53%

C.55%

D.57%

答案：1-5:CCCBC

六、截位直除：一步除法和多步除法

◆ **什么是截位？**

截位：从左边第一个非 0 的数开始截（截几位即保留几位，下一位数四舍五入）

0.17258（保留两位有效数字）_____

0.17258（保留三位有效数字）_____

◆ **截谁？看算式形式**

（1）一步除法：直截分母

常见形式：$\frac{A}{B}$、$\frac{A+B}{C}$、$\frac{A}{B+C}$

多步除法：分子分母都截（截完约分）

常见形式：$\frac{A}{B}\times\frac{C}{D}$、$\frac{A}{B}\div C$

◆ **截几位？　　看选项差距**

1.选项差距大（截两位）

①首位均不同

②首位相同，第二位不同，第二位差＞首位

2.选项差距小（截三位）：

首位相同，第二位不同，第二位差≤ 首位

注意：

1. 选项为某整数的左右邻居，选项差距小，截三位（如：59、61）

2.选项差距极小：首位相同，第二位也相同（需要精算，截四位）

（1）一步除法

【例 1】$\frac{6735}{4489}$

A.1.5

B.2.3

C.3.2

D.4.1

【例 2】$\frac{14.35}{54.89}$

A.22.1%

B.26.1%

C.29.2%

D.45.7%

【例 3】$\frac{32456.32}{1452}$

A.21.64

B.22.35

C.24.52

D.26.31

【例 4】$\frac{31328.1}{1+13.1\%}$

A. 27699.5

B. 27712.2

C. 28233.6

D. 31328.1

【例 5】$\frac{7635.2}{1375.1+210.2+532.6}$

A.1.3

B.2.4

C.3.6

D.4.3

【例 6】$\frac{3734.39-2912.62}{5250.4}$

A.12.4%

B.15.7%

C.22.3%

D.32.5%

（2）多步除法

【例 7】$\frac{13695}{2357}\times\frac{2992}{21284}$

A.0.6

B.0.8

C.1.2

D.1.6

【例 8】$\frac{48352.1}{32161.9}\times\frac{6926.7}{2314.7}$

A.4.5

B.6.6

C.8.3

D.10.2

【例 9】$\frac{5834.5}{2341}\times\frac{4340.2}{1102}$

A.7.5

B.9.8

C.11.4

D.17.3

七、截位直除：量级

若选项之间存在约 10、100 倍的关系时：

方法：

① 截位（（组与组的差距）差距大截两位、差距小截三位）

② 保留量级（（组内关系）小数点位置：决定是几十还是几百）

根据选项判定是否有量级：

【例 10】$\frac{7036.7}{12.5\%}$

A.52324

B.54063

C.56294

D.58375

【例 11】$\frac{7036.7}{12.5\%}$

A.4.8 万

B.48 万

C.5.6 万

D.56 万

【例 12】$\frac{8545.6亿}{241123.4万}\approx$

A.354

B.3540

C.364

D.3640

易错点：量级牵扯到百分号（%） **方法：分母划线，两位即是%**

【例 13】$\frac{534}{24189}$

A.22%

B.2.2%

C.45%

D.4.5%

【例 14】$\frac{2013.89-1989.61}{4750.9}$

A.0.5%

B.1.7%

C.3.5%

D.5.1%

答案：1-5:ABBAC；6-10:BBABC；11-14：CABA

八、分数比较

【例 1】大小比较：$\frac{4563}{721.6}$　$\frac{4247}{743.5}$

【例 2】大小比较：$\frac{27.06}{43.14}$　$\frac{60.31}{80.54}$

【例 3】大小比较：$\frac{27.06}{43.14}$　$\frac{60.31}{130.54}$

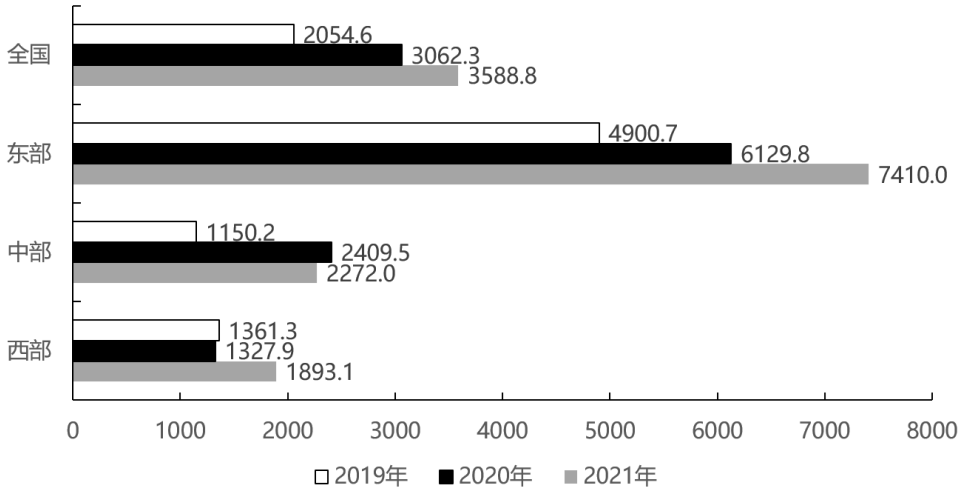


图1 2019~2021年县域农业农村信息化建设县均社会资本投入（单位：万元）

【例 4】（2024 广东）根据资料，判断正误（　　）。

2019~2021 年，中部地区县均社会资本投入增速高于全国县均社会资本投入增速

$$\text{中部}\frac{2272.0}{1150.2}>\text{全国}\frac{3588.8}{2054.6}$$

A.正确　B.错误

【例 5】$\frac{46782}{180.23}$、$\frac{25487}{132.77}$、$\frac{45669}{119.70}$、$\frac{39642}{139.05}$，最大的是

A.$\frac{46782}{180.23}$

B.$\frac{25487}{132.77}$

C.$\frac{45669}{119.70}$

D.$\frac{39642}{139.05}$

【例 6】$\frac{624.13}{41.23}$、$\frac{425.24}{57.24}$、$\frac{726.75}{74.06}$、$\frac{376.22}{78.54}$，最小的是

A.$\frac{624.13}{41.23}$

B.$\frac{425.24}{57.24}$

C.$\frac{726.75}{74.06}$

D.$\frac{376.22}{78.54}$

【例 7】①$\frac{5674}{2243}$、②$\frac{3526}{1824}$、③$\frac{4687}{3564}$、④$\frac{4214}{3695}$，从高到底排序，正确的是

A.②①③④

B.①②③④

C.②①④③

D.①②④③

【例 8】①$\frac{22.24}{13947}$、②$\frac{2.45}{9060}$、③$\frac{8.8}{12080}$、④$\frac{9.2}{6680}$从高到底排序，正确的是

A.②>①>③

B.①>③>④

C.②>①>④

D.①>④>③

【例 9】2017—2021 年，哪两年高于$\frac{654.91}{262}$

2017 年：$\frac{22.91}{58}$

2018 年：$\frac{100.7}{45}$

2019 年：$\frac{230.8}{34}$

2020 年：$\frac{70.9}{33}$

2021 年：$\frac{207}{77}$

A.2018 年和 2019 年

B.2018 年和 2020 年

C.2019 年和 2020 年

D.2019 年和 2021 年

【例 10】 $\frac{9245}{13.82}$、$\frac{31379}{74.06}$、$\frac{441}{2.36}$、$\frac{3105}{39.08}$这四个分数最大的是？

A. $\frac{9245}{13.82}$

B. $\frac{31379}{74.06}$

C. $\frac{441}{2.36}$

D. $\frac{3105}{39.08}$

答案：1-5:＞、＜、＞、AC；6-10: DBDDA

第二章：基期量

➢ 作为对比参照的是基期，而相对于基期比较的是现期

现期、基期、增长量、增长率

2024 年 12 月我的工资 120 元，2023 年 12 月我的工资 100 元。

2024 年 12 月比 2023 年 12 月同比增长了 20 元（增长量）

2024 年 12 月比 2023 年 12 月同比增长 20%（增长率：r）

同比和环比的区分：（_____、_____）

同比：与上年同期相比

环比：与紧紧相邻的上一统计周期相比（月环比、季度环比）

【练一练】

	同比	环比
2024 年 9 月		
2024 年第三季度		
2024 年第一季度		

公式：

（1）

（2）

1、基期计算

加减计算

8月份，煤炭进口2868万吨，比上月下降33万吨，但仍保持较高水平，同比增长13.5%。1—8月份，煤炭进口2.0亿吨，同比增长14.7%。

【例1】（2021浙江）本年度上半年，煤炭进口量约为多少万吨？

A.14198

B.14231

C.14264

D.14297

2022 年末全国参加城镇职工基本养老保险人数 50349 万人，比上年末增加 2275 万人。参加城乡居民基本养老保险人数 54952 万人，比上年末增加 155 万人；参加失业保险人数 23807 万人，比上年末增加 849 万人；参加工伤保险人数 29111 万人，比上年末增加 825 万人，参加生育保险人数 24608 万人，比上年末增加 856 万人。

【例 2】（2023 江苏事业单位）2021 年末，全国参加失业保险人数比参加工伤保险人数少多少？（ ）

A.5280 万人

B.5304 万人

C.5328 万人

D.5426 万人

除法：r>0、r<0

公式：基期量=$\frac{\text{现期量}}{1+r}$

方法：分析选项，截位直除

2023 年一季度，新疆外贸进出口总值 680.7 亿元，同比增长 80.3%。其中，出口 584.7 亿元，同比增长 86.9%。3 月当月，新疆外贸进出口总值 236.9 亿元，同比增长 70%。其中，出口 203.4 亿元，同比增长 78.9%；进口 33.5 亿元，同比增长 30.8%。

【例 3】（2024 联考）2022 年 3 月，新疆外贸出口值约为：

A.126 亿元

B.114 亿元

C.139 亿元

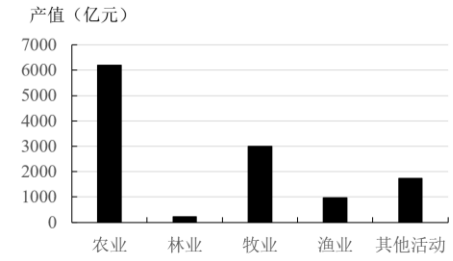
D.160 亿元

表1 2022年农林牧渔业产值及增长速度

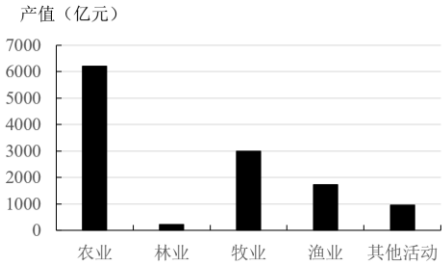
指标	产值（亿元）	比上年增长（%）
农林牧渔业	12130.7	4.8
农业	6206.5	4.7
林业	227.3	8.5
牧业	3003.5	3.5
渔业	1729.7	4.4
农林牧渔专业及辅助性活动	963.7	9.1

【例 4】（2024 山东）下列各图符合 2021 年农林牧渔业各组成部分产值数量关系的是:

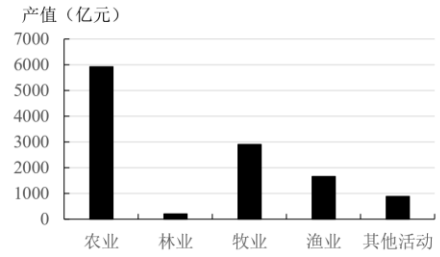
A.



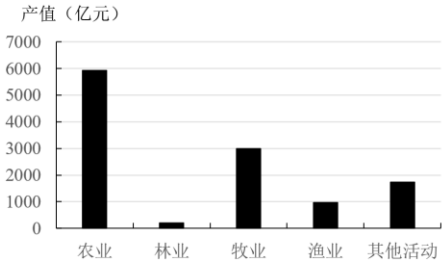
B.



C.



D.



2022 年一季度部分省市软件和信息技术服务业完成情况

	软件业务收入		其中：信息技术服务收入	
	本年累计（亿元）	同比增长（%）	本年累计（亿元）	同比增长（%）
全国	20059.67	11.6	13102.09	13.7
贵州	144.3	100.8	133.14	101.4
北京	4394.68	13.2	2894.97	13.9
天津	420.08	2.1	273.05	1.3
河北	84.18	20.3	73.43	21.7
上海	1647.5	17.6	1126.03	22.0
江苏	2805.58	10.5	1697.25	11.7
浙江	1725.91	5.4	1357.55	5.3
福建	414.63	15.7	231.61	17.5
山东	1274.51	18.0	618.44	24.8
广东	3785.08	6.1	2610.26	10.2

【例 5】（2024 浙江）2021 年一季度，浙江软件业务收入累计额约为多少亿元?

A.1600

B.1640

C.1680

D.1800

（2023 年前三季度）分行业类别看，新闻信息服务实现营业收入 12203 亿元，同比增长 14.9%；内容创作生产 19857 亿元，同比增长 10.3%；创意设计服务 14973 亿元，同比增长 9.6%；文化传播渠道 10712 亿元，同比增长 12.4%；文化投资运营 473 亿元，同比增长 27.7%；文化娱乐休闲服务 1289 亿元，同比增长 67.2%；文化生产和中介服务 10894 亿元，同比下降 1.5%；文化装备生产 4443 亿元，同比下降 4.3%；文化消费终端生产 16775 亿元，同比增长 2.1%。

【例 6】（2024 联考）2023 年前三季度，实现营业收入最高的文化行业，其 2022 年前三季度的营业收入大约是：

A.不到 1.4 万亿元

B.1.4~1.8 万亿元

C.1.8~2.2 万亿元

D.超过 2.2 万亿元

表1：2022年1-3月我国对外承包工程业务情况

时间	累计完成营业额（亿元）	同比增速（%）	累计新签合同额（亿元）	同比增速（%）
1月	525.9	-8.2	955.8	-1.2
2月	1141.7	-3.3	1932.1	-2.5
3月	1841.6	-5.7	3009.4	-13.3

【例 7】（2023 全国事业单位联考）2021 年第一季度我国对外承包工程业务新签合同额在以下哪个范围内？（　　）

A.不到 3000 亿元

B.3000 亿元~4000 亿元之间

C.4000 亿元~5000 亿元之间

D.超过 5000 亿元

2021 年 1-5 月，全国共破获电信网络诈骗案件 11.4 万起，打掉犯罪团伙 1.4 万个，抓获犯罪嫌疑人 15.4 万名，同比分别上升 60.4%、80.6%和 146.5%。2021 年 5 月，全国共立电信网络诈骗案件 8.46 万起，与 4 月相比下降 14.3%。

【例 8】（2022 国考）2021 年 4-5 月，全国共立电信网络诈骗案件约多少万起？

A.12

B.14

C.16

D.18

化除为乘

2020年H省秋粮平均生产成本及同比增速

单位：元/亩（成本），%（增速）

	秋粮		玉米		稻谷	
	成本	增速	成本	增速	成本	增速
生产成本	440.6	-2.1	430.5	-1.9	525.7	-4.0
其中：物质费用	203.9	-0.3	205.4	-0.3	210.6	0.4
其中：种子	51.3	0.4	48.5	-0.1	68.4	1.8
化肥	125.3	-2.2	131.0	-2.6	107.7	-0.3
农药	26.9	8.6	25.6	14.0	34.3	-1.2
生产服务支出	130.3	-3.5	120.7	-5.8	194.4	4.3
其中：机耕	22.1	0.3	15.4	-3.1	61.8	3.0
机播	21.7	-1.1	21.2	-6.4	24.0	37.4
机收	62.3	-2.0	61.0	-1.2	78.4	-0.7
排灌	24.3	-12.0	23.1	-17.2	30.2	0.7
人工成本	106.5	-3.5	104.4	-0.4	120.7	-20.3

注：部分数据因四舍五入的原因，存在总计与分项合计不等的情况。

【例 9】（2022 国考）2019 年，H 省秋粮稻谷的平均生产成本约为多少元/亩？

A.439

B.450

C.533

D.548

2022 年，全国居民人均可支配收入 36883 元，比上年增长（以下如无特别说明，均为同比名义增长）5.0%。分城乡看，城镇居民人均可支配收入 49283 元，增长 3.9%；农村居民人均可支配收入 20133 元，增长 6.3%。

【例 10】（2023 广东）2021 年，全国居民人均可支配收入约为（　　）万元。

A.3.33

B.3.42

C.3.51

D.3.60

化除为乘反例

化除为乘不可用：

（1）选项首位相同，第二位也相同 且 （2）$4\% \leq |r| \leq 5\%$。

2017 年全区粮食种植面积 3433 万亩，比上年下降 5%。其中，小麦种植面积 1792 万亩，下降 7.3%；玉米种植面积 1369 万亩，下降 0.6%。棉花种植面积 3326 万亩，增长 2.9%。油料种植面积 361 万亩，增长 0.1%。甜菜种植面积 95 万亩，下降 5.0%。

【例 11】（2018 新疆）2016 年全年该地区全区粮食种植面积约是多少万亩？

A.3654

B.3635

C.3603

D.3614

2、基期比较

2022年中国对外直接投资流量行业分布

行业	流量（亿美元）	增长（%）
合计	1631.2	-8.8
租赁和商务服务业	434.8	-11.9
制造业	271.5	1.0
金融业	221.2	-17.5
批发和零售业	211.7	-24.8
采矿业	151.0	79.5
交通运输/仓储和邮政业	150.4	23.0
电力/热力/燃气及水的生产和供应业	54.5	24.1
科学研究和技术服务业	48.2	-4.9
房地产业	22.1	-46.1
信息传输/软件和信息技术服务业	16.9	-67.1
文化/体育和娱乐业	15.3	1600.0
建筑业	14.5	-68.6
居民服务/维修和其他服务业	6.8	-62.4
农/林/牧/渔业	5.1	-45.2
卫生和社会工作	2.9	-14.7
教育	2.4	700.0
水利/环境和公共设施管理业	1.8	-18.2
住宿和餐饮业	0.1	-96.3

【例 12】（2024 广东）2021 年，中国下列行业的对外直接投资流量，从大到小排序正确的是（　）。

A.金融业>租赁和商务服务业>批发和零售业>制造业

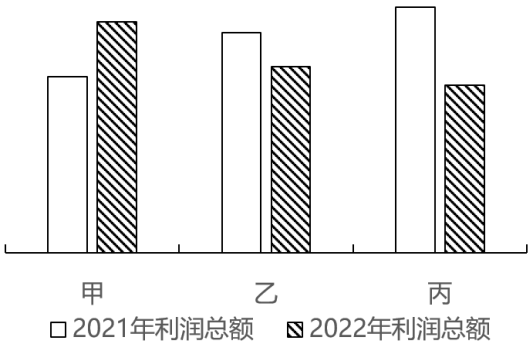
B.金融业>租赁和商务服务业>制造业>批发和零售业

C.租赁和商务服务业>制造业>批发和零售业>金融业

D.租赁和商务服务业>批发和零售业>制造业>金融业

2022 年，在 41 个工业大类行业中，利润总额由高到低的前十个行业的利润情况如下：煤炭开采和洗选业实现利润总额 10202 亿元，同比增长 44.3%；计算机、通信和其他电子设备制造业实现利润总额 7389.5 亿元，同比下降 13.1%；化学原料和化学制品制造业实现利润总额 7302.6 亿元，同比下降 8.7%；电气机械和器材制造业实现利润总额 5915.6 亿元，同比增长 31.2%；汽车制造业实现利润总额 5319.6 亿元，同比增长 0.6%；非金属矿物制品业实现利润总额 4759 亿元，同比下降 15.5%；医药制造业实现利润总额 4288.7 亿元，同比下降 31.8%；石油和天然气开采业实现利润总额 3545 亿元，同比增长 109.8%；通用设备制造业实现利润总额 3250.3 亿元，同比增长 0.4%；电力、热力生产和供应业实现利润总额 3154 亿元，同比增长 86.3%。

【例 13】（2023 联考）下面柱状图中，甲、乙、丙依次代表（　）。



A.电气机械和器材制造业、非金属矿物制品业、医药制造业

B.化学原料和化学制品制造业、电气机械和器材制造业、汽车制造业

C.汽车制造业、电气机械和器材制造业、化学原料和化学制品制造业

D.医药制造业、非金属矿物制品业、电气机械和器材制造业

3、基期和差

2022年交通固定资产投资额及同比增长率

	交通固定资产投资额（亿元）	同比增长率（%）
铁路	7109	-5.1
公路	28527	9.7
其中：高速公路	16262	7.3
普通国省道	5973	6.5
农村公路	4733	15.6
水路	1679	10.9
其中：内河建设	867	16.7
沿海建设	794	9.9
民航	1231	0.7

【例 14】（2024 联考）2021 年公路交通固定投资额与水路交通固定投资额共（　）。

A.不到 2 万亿元

B.2~3 万亿元

C.3~4 万亿元

D.4 万亿元以上

截至 2022 年底，全国共有 278 家银行机构和 29 家理财公司有存续的理财产品，共存续产品 3.47 万只，同比下降 4.41%；存续规模 27.65 万亿元，同比下降 4.66%。分机构类型来看，理财公司存续产品 13947 只，存续规模 22.24 万亿元，同比增长 29.36%。银行机构中，城商行存续产品 9064 只，存续规模 2.45 万亿元，同比下降 32.34%；农村金融机构存续产品 7808 只，存续规模 1.09 万亿元，同比下降 2.63%；股份制银行存续产品 1208 只，存续规模 0.88 万亿元，同比下降 82.99%；大型银行存续产品 668 只，存续规模 0.92 万亿元，同比下 49.26%；其他机构存续产品 1980 只，存续规模 0.07 万亿元，同比下降 13.42%。

【例 15】（2024 联考）2021 年银行机构存续的理财产品存续规模约为多少万亿元？

A.5.4

B.6.7

C.10.5

D.11.8

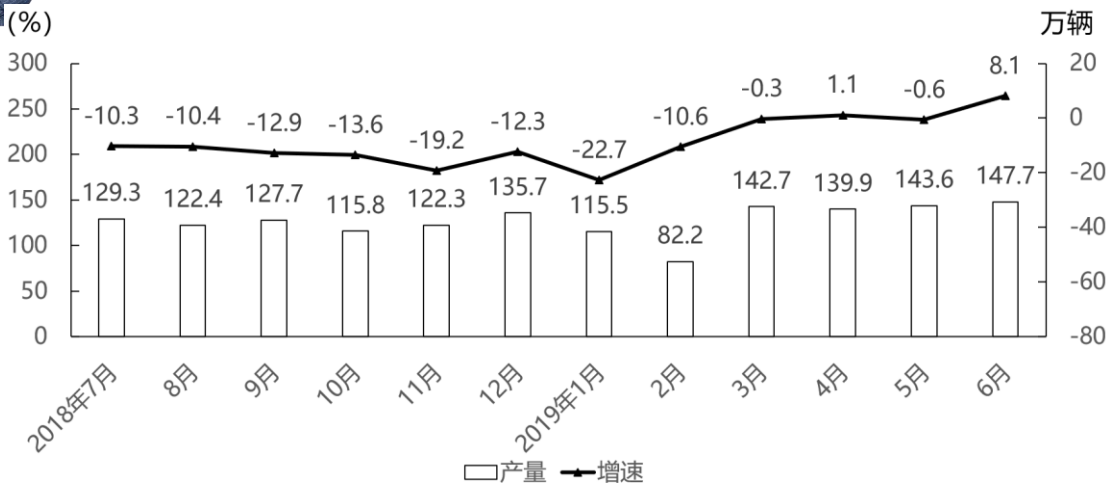


图1 2018年7月~2019年6月我国摩托车产量及同比增速

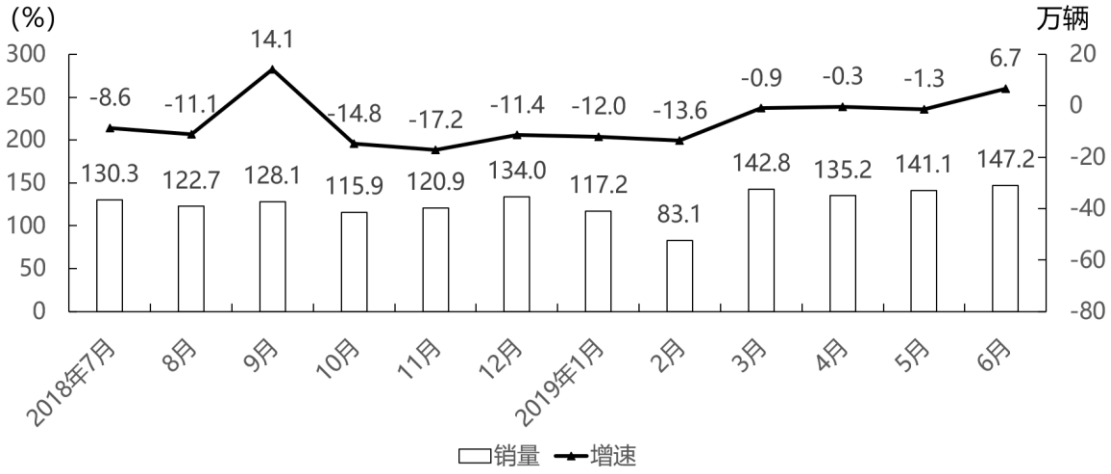


图2 2018年7月~2019年6月我国摩托车销量及同比增速

【例 16】（2024 四川）2018 年 1 月，我国摩托车产量比销量：

A.少 10 万辆以内

B.少 10 万辆以上

C.多 10 万辆以内

D.多 10 万辆以上

2022年6月全国各类型彩票销售情况表

类型	6月			1~6月累计	
	销售额(亿元)	同比增长(%)	环比增长(%)	销售额(亿元)	同比增长(%)
一、福利彩票	130.8	16.5	-0.5	748.6	10.6
（一）乐透数字型	71.8	3.7	-8.3	433.1	-6.0
（二）即开型	28.3	30.3	11.8	175.6	33.2
（三）基诺型	30.7	44.2	10.2	139.9	65.5
二、体育彩票	189.0	-19.2	-3.7	1072.0	-3.2
（一）乐透数字型	58.6	5.0	-5.0	330.9	-13.0
（二）竞猜型	104.5	-33.1	-4.1	580.0	-3.0
（三）即开型	25.9	18.5	1.4	161.1	24.7
（四）视频型	0.0015	143.4	355.1	0.0046	-25.9

注：全国彩票销售额为福利彩票销售额与体育彩票销售额之和

【例 17】（2024 浙江）2021 年上半年全国彩票销售额约为多少亿元?

A.1780

B.1810

C.1840

D.1880

4、变形公式的运用一

$$\text{基期}=\frac{\text{增长量}}{r}$$

从棉区看，2016 年黄河、长江流域棉区延续 2015 年减产较多的趋势。其中，黄河流域棉花播种面积减少 147.8 千公顷，下降约 14.3%；单产每公顷增加 63.3 公斤，提高约 6.0%；产量减少 10.0 万吨，下降约 9.2%。长江流域棉花播种面积减少 160.7 千公顷，下降约 19.8%；单产每公顷减少 68.3 公斤，下降约 5.9%；产量减少 23.0 万吨，下降约 24.6%。

【例 18】（2021 四川下）2015 年，黄河流域的棉花单产为：

A.1118 公斤/公顷

B.1092 公斤/公顷

C.1055 公斤/公顷

D.1003 公斤/公顷

2019 年，全国棉花产量 588.9 万吨，比上年减少 21.3 万吨。其中，新疆棉花产量 500.2 万吨，比上年减少 10.8 万吨；全国棉花种植面积为 3339.2 千公顷，比上年减少 15.2 千公顷。新疆的棉花种植面积比上年增加 49.2 千公顷。长江流域棉花种植面积比上年减少 32.4 千公顷，同比下降 8.7%。黄河流域棉花种植面积比上年减少 28.1 千公顷，同比下降 6.2% 。

【例 19】（2021 山东）2018 年长江流域棉花种植面积约是黄河流域棉花种植面积的多少倍？

A.0.5

B.0.8

C.1.2

D.2.1

5、变形公式的运用二

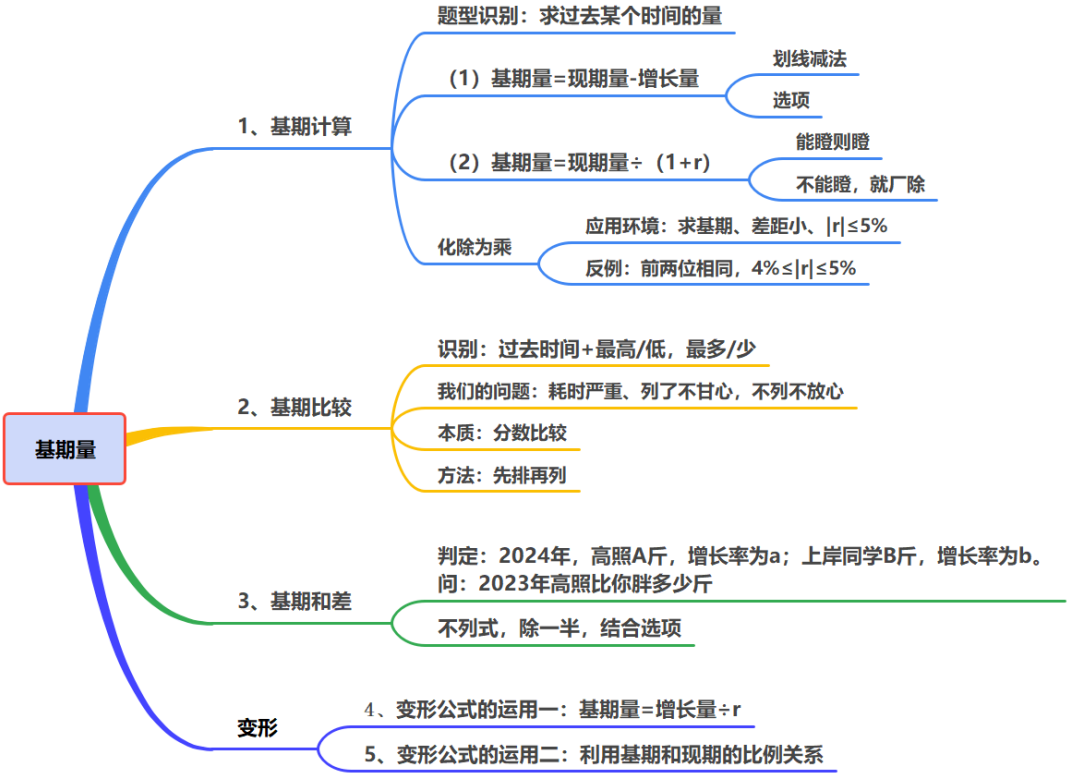
2023 年二季度，全国跨省异地就医直接结算 2854.13 万人次，.....环比分别增长 46.02%

【例 20】（2024 江苏）2023 年，二季度全国跨省联网异地就医费用直接结算人次占上半年的比重为（ ）

A.57.1%

B.59.3%

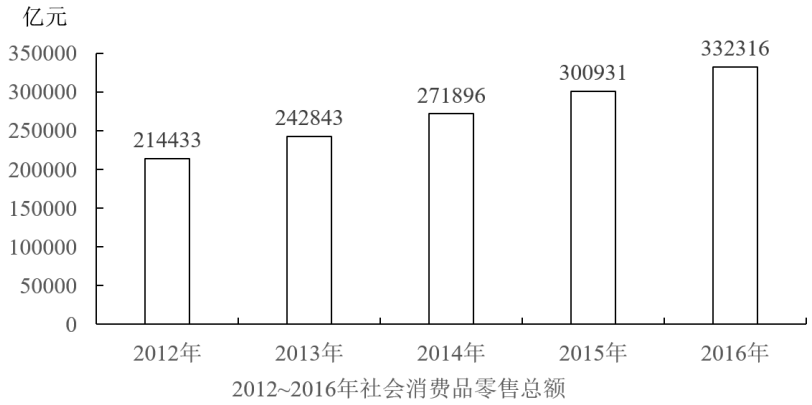
答案：1-5:BCBCB；6-10：CBDDC；11-15:DDABD；16-20:DACBB



第三章：现期量

1、保持增长量

（1）求具体值



2012~2016年社会消费品零售总额

【例 1】（2021 四川下）如从 2016 年开始，社会消费品零售总额年增量保持不变，社会消费品零售总额首次超过 40 万亿元的年份是：

A.2017 年

B.2018 年

C.2019 年

D.2020 年

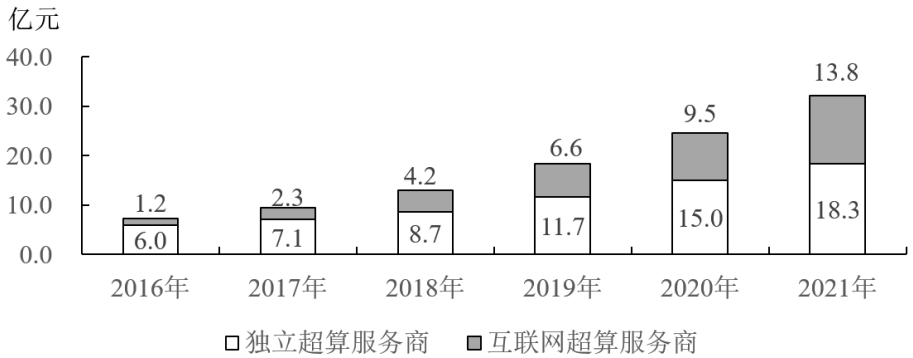
（2）现期追赶

考法一：小追大，追上，问几年。

方法：找差距，补差距。

考法二：小追大，追不上，问差距。

考法一：小追大，追上，问几年。



2016~2021年中国第三方超算服务市场规模

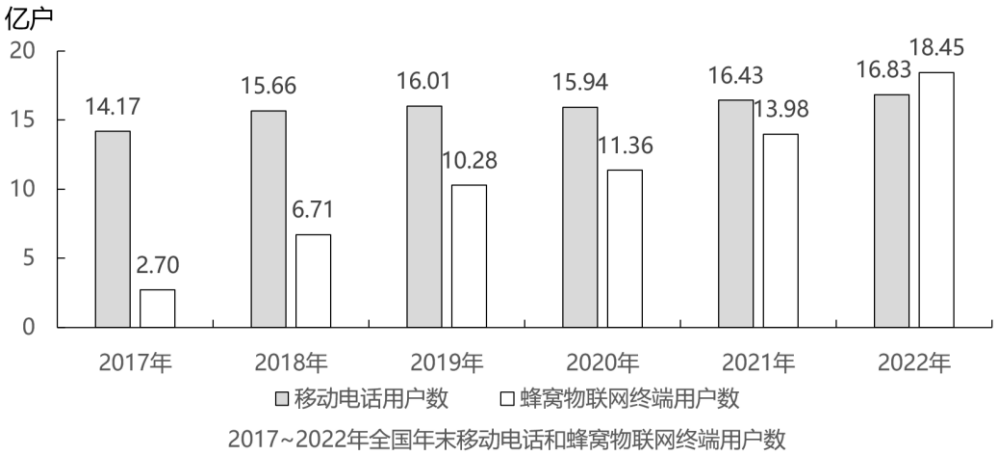
【例 2】（2023 国考）如保持 2021 年同比增量不变，则到哪一年第三方互联网超算服务商提供的服务市场规模将第一次超过第三方独立超算服务商？

A.2025 年

B.2026 年

C.2027 年

D.2028 年



2017~2022年全国年末移动电话和蜂窝物联网终端用户数

【例 3】（2024 国考）如 2022 年内每个月移动电话用户数增量和蜂窝物联网终端用户数增量均为固定值，则 2022 年蜂窝物联网终端用户数第一次超过移动电话用户数是在哪个季度？

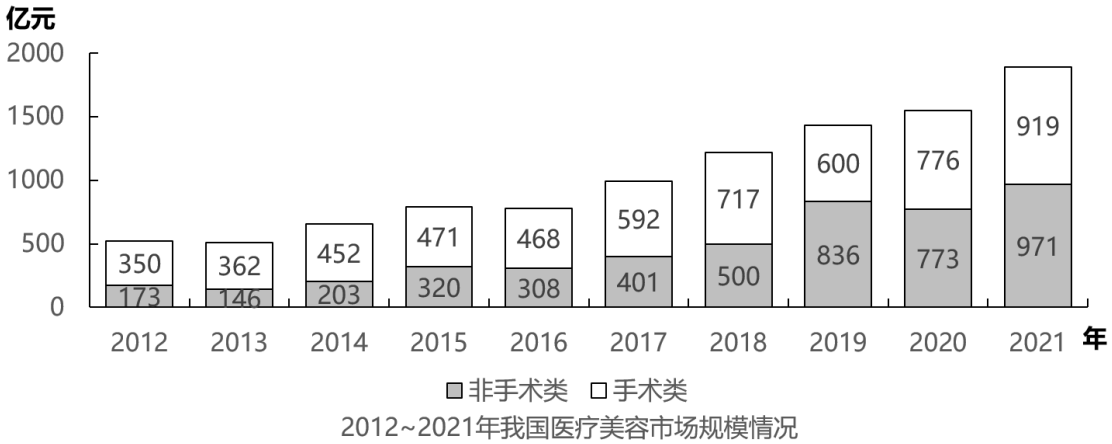
A.第一季度

B.第二季度

C.第三季度

D.第四季度

考法二：小追大，追不上，问差距。



2012~2021年我国医疗美容市场规模情况

【例 4】（2023 全国事业单位联考）如 2022 年后每年的同比增量均与 2021 年的同比增量一致，则到 2025 年，我国非手术类医疗美容市场规模与手术类医疗美容市场规模的差值在以下哪个范围内？（　）

A.不到 250 亿元

B.250 亿元-260 亿元之间

C.260 亿元-270 亿元之间

D.270 亿元以上

2、保持增长率

（1）给增长率，保持增长率

估算：3224×（1+12.6%）≈3224+3224×12.5%=3224+3224×$\frac{1}{8}$

精算：3224×（1+12.6%）=3224+3224×（12.5%+0.1%）=3224+3224×（$\frac{1}{8}$+0.1%）

2017~2021年我国就业基本情况

	2017年	2018年	2019年	2020年	2021年
劳动力（万人）	79042	78653	78985	78392	?
就业人员（万人）	76058	75782	75447	75064	74652
第一产业	20295	19515	18652	17715	17072
第二产业	21762	21356	21234	21543	21712
第三产业	34001	34911	35561	35806	35868
按城乡分就业人员（万人）					
城镇就业人员	43208	44292	45249	46271	46773
乡村就业人员	32850	31490	30198	28793	27879
按登记注册类型（万人）					
国有单位	6064	5740	5473	5563	5633
有限责任公司	6367	6555	6608	6542	6526
外商投资单位	1291	1212	1203	1216	1220
城镇登记失业人员（万人）	972	974	945	1160	1040

【例 5】（2024 天津事业单位）若 2021 年劳动力同比增长率为-0.5%，问号处应填入的数字为（　　）？

A.76500

B.77000

C.77500

D.78000

截至 2018 年底，中国人工智能市场规模约为 238.2 亿元，同比增长率达到 56,6%。从中国人工智能企业地域分布情况来看，北京企业数量最多，企业数量为 368 家；其次为广东，人工智能企业数量为 185 家；排名第三的是上海，数量为 131 家。

【例 6】（2020 联考）若按照 2018 年同比增长率，到 2019 年底中国人工智能市场规模约为：

A.363 亿元

B.371 亿元

C.373 亿元

D.383 亿元

2022 年废有色金属中废铅回收重量同比增长 5.56%。

2021~2022年我国十个品种再生资源回收情况

序号	名称	回收重量（万吨）		回收金额（亿元）	
		2021年	2022年	2021年	2022年
1	废钢铁	25021.0	24081.0	7523.6	6911.2
2	废有色金属	1348.0	1375.0	2878.5	2959.7
3	废塑料	1900.0	1800.0	1050.0	1050.0
4	废纸	6491.0	6585.0	1493.0	1402.6
5	废轮胎	640.0	675.0	76.8	101.3
6	废弃电器电子产品	463.0	415.0	222.4	227.4
7	报废机动车	678.5	820.7	276.9	311.9
8	废旧纺织品	475.0	415.0	26.1	16.6
9	废玻璃	1005.0	850.0	48.0	38.3
10	废电池（铅酸电池除外）	42.0	51.0	99.7	121.6

2021年废有色金属中各类废金属回收重量占比情况

【例 7】（2024 联考）2022 年我国废有色金属中废铅回收重量约为：

A.261 万吨

B.275 万吨

C.285 万吨

D.291 万吨

（2）不给增长率，保持增长率

方法一：r＞0，增长量变大。

方法二：大精小估，利滚利思想（正向：计算出具体值、反向：选项排除）

例子：

城镇居民和职工基本医疗保险参保人数（万人）

年份	合计	城镇居民基本医疗保险	城镇职工基本医疗保险	在岗职工	退休人数
2013年	57073	29629	27443	20501	6942
2014年	59747	31451	28296	21041	7255
2015年	66582	37689	28893	21362	7531
2016年	74392	44860	29532	21720	7812
2017年	117681	87359	30323	22288	8035
2018年	134459	102778	31681	23308	8373
2019年	135407	102483	32925	24225	8700
2020年	136131	101676	34455	25429	9026
2021年	136297	100866	35431	26107	9324
2022年	134570	98328	36242	26607	9635

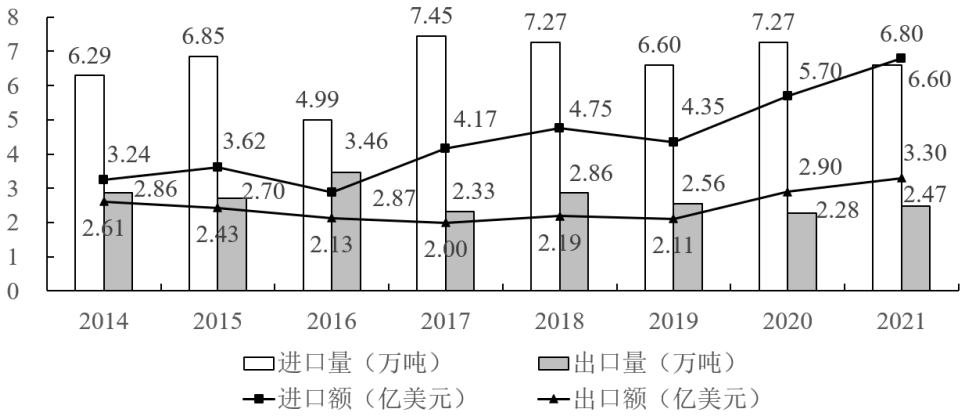
【例 8】（2023 河北事业单位）若从 2023 年起，每年均按 2018-2022 年的平均增速增长，则 2027 年城镇职工基本医疗保险参保人数将达到（　　）。

A.43316 万人

B.41460 万人

C.37878 万人

D.34564 万人



2014~2021年中国农作物种子进出口贸易数据

【例 9】（2022 联考）如按 2021 年我国农作物种子出口量同比增速推算，2022 年我国农作物种子出口量约为多少万吨？

A.2.58

B.2.68

C.2.78

D.2.88

2012~2020年居民医保基金收支情况（单位：亿元）

	2012	2013	2014	2015	2016	2017	2018	2019	2020
基金收入	877	1187	1649	2109	2811	5653	6971	（？）	9115
基金支出	675	971	1437	1781	2480	4955	6277	8191	8165
结余率	23.0%	18.2%	12.9%	11.8%	11.8%	12.4%	10.0%	4.5%	10.4%

【例 10】（2022 联考）假设 2021 年居民医保基金收入同比增速与 2020 年相同，那么，2021 年居民医保基金收入约为：（表格？为 8577）

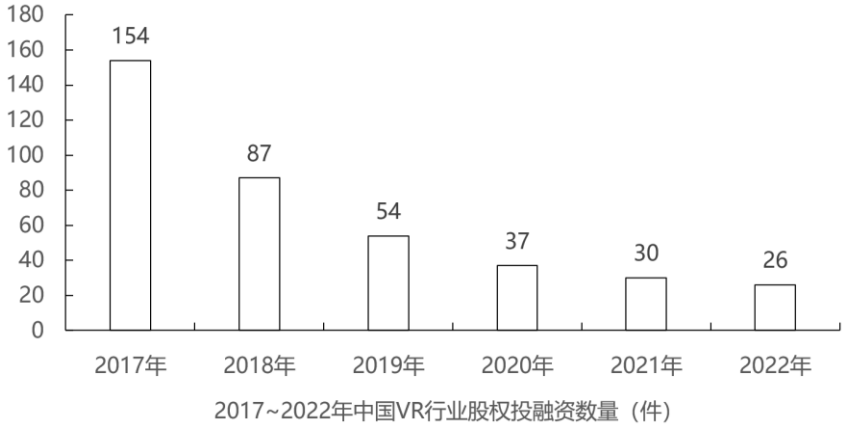
A.9598 亿元

B.9689 亿元

C.9727 亿元

D.9873 亿元

（3）变形考法：负增长



2017~2022年中国VR行业股权投融资数量（件）

【例 11】（2024 天津事业单位）如果保持 2022 年中国 VR 行业融资数量的同比增长率不变，预计 1 年后融资数量达到约（　　）件。

A.20

B.23

C.26

D.29

3、按照名义增长率求现期

名义增长率：没有扣除价格因素影响得到的增长率 实际增长率（按可比价格计算）：扣除价格因素影响得到的增长率 公式 1： 公式 2：

2018 年前三季度，全国居民人均消费支出 14281 元，比上年同期名义增长 8.5%格因素，实际增长 5.3%。

【例 12】（2018 江西法检）2018 年前三季度，消费价格指数（CPI）增长了（　）

A.6.3%

B.4.3%

C.3.3%

D.2.2%

【例 13】（2018 湖北选调）去年小李年工资收入 4 万元，今年预计增加 30%。今年居民消费价格指数（CPI）预计增长 3%，据此推算，今年小李的实际购买力约增加：

A.20.8%

B.25%

C.26.2%

D.35.1%

2020 年全国居民人均消费支出 21210 元，比上年下降 1.6%，扣除价格因素，实际下降 4.0%。其中，人均服务性消费支出 9037 元，比上年下降 8.6%，占居民人均消费支出的比重为 42.6%。按常住地分，城镇居民人均消费支出 27007 元，下降 3.8%，扣除价格因素，实际下降 6.0%：农村居民人均消费支出 13713 元，增长 2.9%。扣除价格因素，实际下降 0.1%。全国居民恩格尔系数为 30.2%。其中城镇为 29.2%。农村为 32.7%。

【例 14】（2022 安徽事业单位）2020 年农村居民消费价格（　　）。

A.同比下降了

B.同比上升了

C.与上年持平

D.呈下降趋势

2019 年一季度，社会消费品零售总额 97790 亿元，同比名义增长 8.3%（扣除价格因素实际增长 6.9%，以下除特殊说明外均为名义增长）。其中，3 月份社会消费品零售总额 31726 亿元，同比增长 8.7%。

【例 15】（2019 河北）按照 2018 年一季度价格计算 2019 年一季度社会消费品零售总额约为多少亿元？

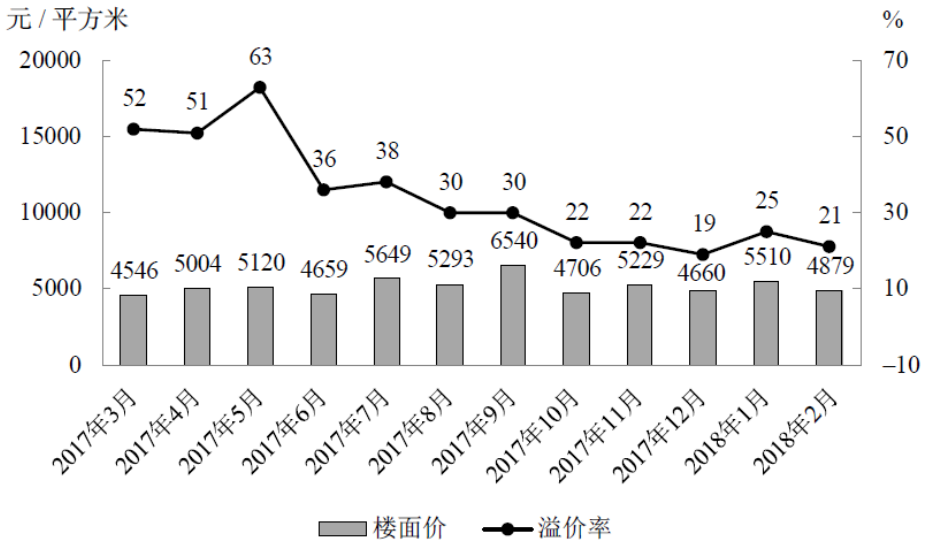
A.85065

B.96526

C.99283

D.114000

4、变形考法



2017 年 3 月—2018 年 2 月全国住宅用地楼面价及溢价率

注：楼面价 = 成交土地总价 / 成交土地规划建筑面积

溢价率 =（住宅销售价格 – 楼面价）/ 楼面价

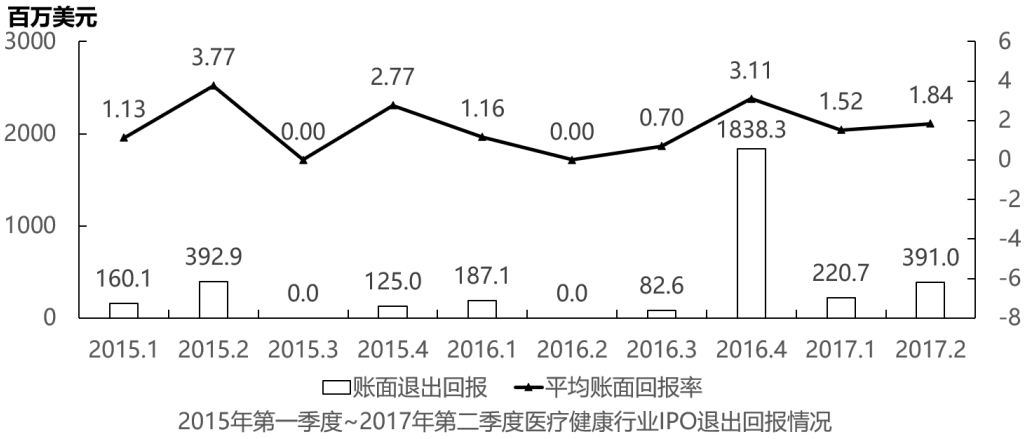
【例 16】（2019 四川）2017 年 3～6 月间，全国住宅销售价格最贵的月份是：

A.3 月

B.4 月

C.5 月

D.6 月



注：平均账面回报率=$\frac{\text{账面退出回报-投资总额}}{\text{投资总额}}$

【例 17】（2019 全国事业单位联考）2017 年第 2 季度医疗健康行业所有 IPO 退出项目的投资总额约为多少亿美元？

A.0.63

B.0.92

C.1.38

D.2.11

答案：1-5:CBCDD；6-10：CCABB； 11-15:BDCBB 16-17:CC