

Обработка последовательностей: модели внимания и трансформеры

К. В. Воронцов
vokov@forecsys.ru

Этот курс доступен на странице вики-ресурса
<http://www.MachineLearning.ru/wiki>
«Машинное обучение (курс лекций, К.В.Воронцов)»

МФТИ • 10 марта 2023

1 Задачи обработки последовательностей

- Рекуррентная сеть
- Рекуррентная сеть с моделью внимания
- Прикладные задачи

2 Разновидности моделей внимания

- Разновидности функций сходства
- Многомерное и иерархическое внимание
- Модель внимания на графах GAT

3 Трансформеры

- Архитектура трансформера
- Трансформер для машинного перевода
- Трансформер BERT

Напоминание. Рекуррентная сеть (RNN)

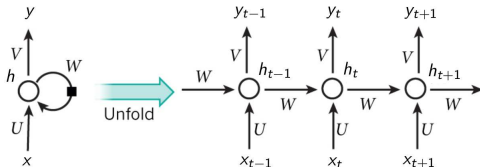
x_t — входной вектор в момент $t = 1, \dots, T$

h_t — вектор скрытого состояния в момент t

y_t — выходной вектор (в некоторых приложениях $y_t \equiv h_t$)

$$h_t = \sigma_h(Ux_t + Wh_{t-1})$$

$$y_t = \sigma_y(Vh_t)$$



Обучение рекуррентной сети: $\sum_{t=0}^T \mathcal{L}_t(U, V, W) \rightarrow \min_{U, V, W}$

- длины входного и выходного сигнала обязаны совпадать
- невозможно заглядывание вперёд
- не подходит для многих задач (MT, QA и др.)

Рекуррентная сеть для синтеза последовательностей (seq2seq)

$X = (x_1, \dots, x_n)$ — входная последовательность

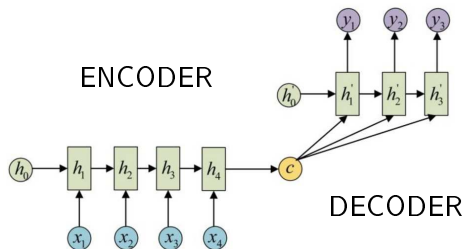
$Y = (y_1, \dots, y_m)$ — выходная последовательность

$c \equiv h_n$ кодирует всю информацию про X для синтеза Y

$$h_i = f_{in}(x_i, h_{i-1})$$

$$h'_t = f_{out}(h'_{t-1}, y_{t-1}, c)$$

$$y_t = f_y(h'_t, y_{t-1})$$



- h_n лучше помнит конец последовательности, чем начало
- чем больше n , тем труднее упаковать всю информацию в c
- придётся контролировать затухание/взрывы градиента
- RNN трудно распараллеливается

Рекуррентная сеть с вниманием (attention mechanism)

$a(h, h')$ — функция сходства состояний входа h и выхода h'

α_{ti} — важность входа i для выхода t (attention score), $\sum_i \alpha_{ti} = 1$

c_t — вектор входного контекста для выхода t (context vector)

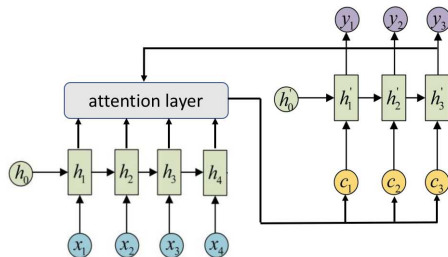
$$h_i = f_{in}(x_i, h_{i-1})$$

$$\alpha_{ti} = \text{norm}_i a(h_i, h'_{t-1})$$

$$c_t = \sum_i \alpha_{ti} h_i$$

$$h'_t = f_{out}(h'_{t-1}, y_{t-1}, c_t)$$

$$y_t = f_y(h'_t, y_{t-1}, c_t)$$



здесь и далее $\text{norm}_i(p_i) = \frac{p_i}{\sum_k p_k}$

- можно отказаться от рекуррентности как по h_i , так и по h'_t
- можно вводить обучаемые параметры в a и c

Применения моделей внимания

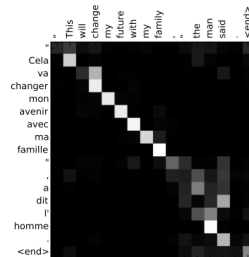
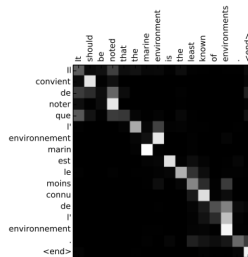
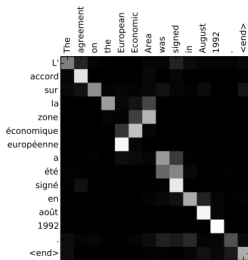
Преобразование одной последовательности в другую, seq2seq:

- Машинный перевод (machine translation)
- Ответы на вопросы (question answering)
- Суммаризация текста (text summarization)
- Описание изображений, аудио, видео (multimedia description)
- Распознавание речи (speech recognition)
- Синтез речи (speech synthesis)

Обработка последовательности:

- Классификация текстовых документов
- Анализ тональности документа / предложений / аспектов

Применения моделей внимания в машинном переводе



Интерпретируемость моделей внимания:

При обработке конкретной последовательности x визуализация матрицы α_{tj} показывает, на какие слова x_j модель обращает внимание, генерируя слово перевода y_t

Модели внимания на изображениях для генерации описаний



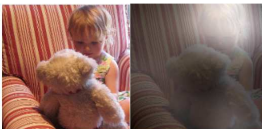
A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

При генерации каждого слова в описании изображения визуализация показывает, на какие области изображения модель обращает внимание, генерируя данное слово

Kelvin Xu et al. Show, attend and tell: neural image caption generation with visual attention. 2016

Разновидности функций сходства векторов

$a(h, h') = h^T h'$ — скалярное произведение

$a(h, h') = \exp(h^T h')$ — тогда norm превращается в SoftMax

$a(h, h') = h^T W h'$ — с матрицей обучаемых параметров W

$a(h, h') = w^T \text{th}(U h + V h')$ — аддитивное внимание с w, U, V

Линейные преобразования векторов query, key, value:

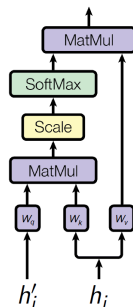
$$a(h_i, h'_{t-1}) = (W_k h_i)^T (W_q h'_{t-1}) / \sqrt{d}$$

$$\alpha_{ti} = \text{SoftMax}_i a(h_i, h'_{t-1})$$

$$c_t = \sum_i \alpha_{ti} W_v h_i$$

$W_q d \times \dim(h')$, $W_k d \times \dim(h)$, $W_v d \times \dim(h)$ — матрицы весов линейных нейронов (обучаемые линейные преобразования в пространство размерности d)

Возможно упрощение модели: $W_k \equiv W_v$



Dichao Hu. An introductory survey on attention mechanisms in NLP problems. 2018.

Формула внимания

q — вектор-запрос, для которого хотим вычислить контекст
 $K = (k_1, \dots, k_n)$ — векторы-ключи, сравниваемые с запросом
 $V = (v_1, \dots, v_n)$ — векторы-значения, образующие контекст
 $a(k_i, q)$ — оценка релевантности (сходства) ключа k_i запросу q
 c — искомый вектор контекста, релевантный запросу

Модель внимания — это 3x-слоенная сеть, вычисляющая выпуклую комбинацию значений v_i , релевантных запросу q :

$$c = \text{Attn}(q, K, V) = \sum_i v_i \text{SoftMax}_i a(k_i, q)$$

$c_t = \text{Attn}(W_q h'_{t-1}, W_k H, W_v H)$ — пример с предыдущего слайда, где $H = (h_1, \dots, h_n)$ — входные векторы, h'_{t-1} — выходной

Внутреннее внимание или «самовнимание» (self-attention):

$c_i = \text{Attn}(W_q h_i, W_k H, W_v H)$ — частный случай, когда $h_i \in H$

Многомерное внимание (multi-head attention)

Идея: J разных моделей внимания совместно обучаются выделять различные аспекты входной информации (например, части речи, синтаксис, фразеологизмы):

$$c^j = \text{Attn}(W_q^j q, W_k^j H, W_v^j H), \quad j = 1, \dots, J$$

Варианты агрегирования выходного вектора:

$$c = \frac{1}{J} \sum_{j=1}^J c^j \text{ — усреднение}$$

$$c = [c^1 \dots c^J] \text{ — конкатенация}$$

$$c = [c^1 \dots c^J] W \text{ — чтобы вернуться к нужной размерности}$$

Регуляризация: чтобы аспекты внимания были максимально различны, строки $J \times n$ матриц A , $\alpha_{ji} = \text{SoftMax}_i a(W_k^j h_i, W_q^j q)$, декоррелируются ($\alpha_s^T \alpha_j \rightarrow 0$) и разреживаются ($\alpha_j^T \alpha_j \rightarrow 1$):

$$\|AA^T - I\|^2 \rightarrow \min_{\{W_k^j, W_q^j\}}$$

Zhouhan Lin, Y.Bengio et al. A structured self-attentive sentence embedding. 2017.

Иерархическое внимание (hierarchical attention)

Вложенная структура: слова \in предложения \in документы
 x_{it} — слова $t = 1, \dots, T_i$ в предложениях $i = 1, \dots, L$

Сеть первого (нижнего) уровня, обучение эмбедингов s_i :

$h_{it} = \text{BidirGRU}(W_0 x_{it})$ — GRU для векторизации слов

$u_{it} = \text{th}(W_1 h_{it} + b_1)$ — обучаемое преобразование Key

$s_i = \sum_t h_{it} \text{SoftMax}_t(u_{it}^T q_1)$ — эмбединг предложения, Query q_1

Сеть второго (верхнего) уровня, обучение эмбедингов v :

$h_i = \text{BidirGRU}(s_i)$ — GRU для векторизации предложений

$u_i = \text{th}(W_2 h_i + b_2)$ — обучаемое преобразование Key

$v = \sum_i h_i \text{SoftMax}_i(u_i^T q_2)$ — эмбединг документа, Query q_2

Максимизация правдоподобия для классификации документов:

$$\sum_d \sum_y \ln \text{SoftMax}_y(W_y v + b_y) \rightarrow \max$$

Z. Yang, A. Smola et al. Hierarchical attention networks for document classification. 2016.

Модель внимания Graph Attention Network (GAT)

Задача классификации вершин графа $\langle V, E \rangle$ на классы Y

Обучающие данные: $b_{iy} = [$ вершина i в классе $y]$, $i \in V$, $y \in Y$

h_i — входные векторы признаков вершин $i \in V$

c_i — выходные векторы оценок вершин, $\sigma(c_{iy}) = P(y|i)$

$\mathcal{N}(t)$ — множество соседей вершины $t \in V$, её контекст

Многомерное самовнимание, $j = 1, \dots, J$, на вершину t :

$$c_t = \frac{1}{J} \sum_{j=1}^J \sum_{i \in \mathcal{N}(t)} \underbrace{W^j h_i \text{ SoftMax}_i \text{ LeakyReLU}(u^j W^j h_i + v^j W^j h_t)}_{a(W^j h_i, W^j h_t)}$$

Максимизация правдоподобия (log-loss) по всем параметрам

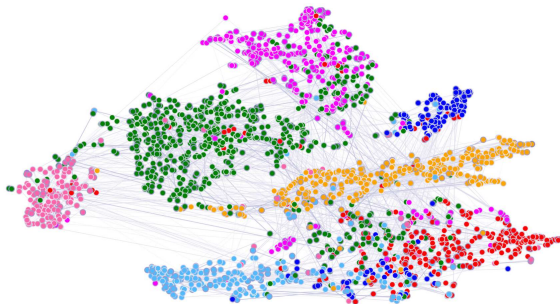
$W = (W^j, u^j, v^j)$ для multi-label классификации вершин графа:

$$\sum_{t \in V} \sum_{y \in Y} b_{ty} \ln \sigma(c_{ty}(W)) + (1 - b_{ty}) \ln \sigma(-c_{ty}(W)) \rightarrow \max_W$$

GAT решает задачи классификации вершин графа

Датасеты Cora, Citeseer, Pubmed для классификации научных статей по словам (признаки h_i) и графу цитирования (рёбра E).

Пример: визуализация векторов c_i с помощью t-SNE, цвета точек — 7 классов, линии — коэффициенты внимания α_{ti}



Petar Veličković et al. Graph Attention Networks. ICLR-2018.

Трансформер для машинного перевода

Трансформер (transformer) — это нейросетевая архитектура на основе моделей внимания и полносвязных слоёв, без RNN

Схема преобразований данных в машинном переводе:

- $S = (w_1, \dots, w_n)$ — слова предложения на входном языке
↓ обучаемая или пред-обученная векторизация слов
- $X = (x_1, \dots, x_n)$ — эмбединги слов входного предложения
↓ трансформер-кодировщик
- $Z = (z_1, \dots, z_n)$ — контекстные эмбединги слов
↓ трансформер-декодировщик, похож на кодировщика
- $Y = (y_1, \dots, y_m)$ — эмбединги слов выходного предложения
↓ генерация слов из построенной языковой модели
- $\tilde{S} = (\tilde{w}_1, \dots, \tilde{w}_m)$ — слова предложения на выходном языке

Vaswani et al. (Google) Attention is all you need. 2017.

Архитектура трансформера-кодировщика

1. Добавляются позиционные векторы p_i :

$$h_i = x_i + p_i, \quad H = (h_1, \dots, h_n) \quad \begin{array}{l} d = \dim x_i, p_i, h_i = 512 \\ \dim H = 512 \times n \end{array}$$

2. Многомерное самовнимание: $j = 1, \dots, J = 8$

$$h_i^j = \text{Attn}(W_q^j h_i, W_k^j H, W_v^j H) \quad \begin{array}{l} \dim h_i^j = 64 \\ \dim W_q^j, W_k^j, W_v^j = 64 \times 512 \end{array}$$

3. Конкатенация:

$$h_i' = \text{MH}_j(h_i^j) \equiv [h_i^{j1} \dots h_i^{jJ}] \quad \dim h_i' = 512$$

4. Сквозная связь + нормировка уровня:

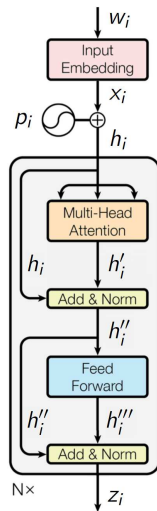
$$h_i'' = \text{LN}(h_i' + h_i; \mu_1, \sigma_1) \quad \dim h_i'', \mu_1, \sigma_1 = 512$$

5. Полносвязная 2х-слойная сеть FFN:

$$h_i''' = W_2 \text{ReLU}(W_1 h_i'' + b_1) + b_2 \quad \begin{array}{l} \dim W_1 = 2048 \times 512 \\ \dim W_2 = 512 \times 2048 \end{array}$$

6. Сквозная связь + нормировка уровня:

$$z_i = \text{LN}(h_i''' + h_i''; \mu_2, \sigma_2) \quad \dim z_i, \mu_2, \sigma_2 = 512$$



Несколько дополнений и замечаний

- вычисления параллельны по элементам последовательности $(x_1, \dots, x_n) \rightarrow (z_1, \dots, z_n)$, что было бы невозможно в RNN
- $N = 6$ блоков $h_i \rightarrow \square \rightarrow z_i$ соединяются последовательно
- возможно использование пред-обученных эмбедингов x_i
- возможно обучение эмбедингов $x_i \in \mathbb{R}^d$ слов $w_i \in V$:
 $x_i = u_{w_i}$ или в матричной записи $X = U \cdot B$, где

$$d \times n \quad d \times |V| \times n$$
 V — словарь слов входных последовательностей,
 U — матрица обучаемых векторных представлений слов,
 $b_{vi} = [w_i = v]$ — матрица бинарного (one-hot) кодирования
- нормировка уровня (Layer Normalization), $x, \mu, \sigma \in \mathbb{R}^d$:

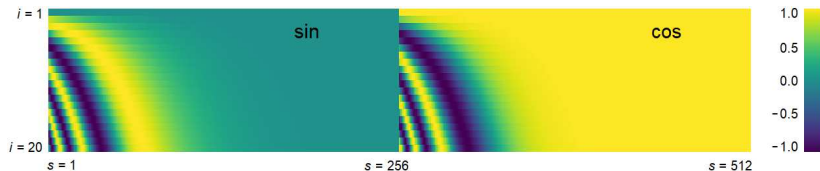
$$\text{LN}_s(x; \mu, \sigma) = \sigma_s \frac{x_s - \bar{x}}{\sigma_x} + \mu_s, \quad s = 1, \dots, d,$$

$$\bar{x} = \frac{1}{d} \sum_s x_s \quad \text{и} \quad \sigma_x^2 = \frac{1}{d} \sum_s (x_s - \bar{x})^2 \quad \text{— среднее и дисперсия } x$$

Позиционное кодирование (positional encoding)

Позиции слов i кодируются векторами p_i , $i = 1, \dots, n$, так, что чем больше $|i - j|$, тем больше $\|p_i - p_j\|$, и n не ограничено:

$$p_{is} = \sin(i 10^{-8 \frac{s}{d}}), \quad p_{i, s + \frac{d}{2}} = \cos(i 10^{-8 \frac{s}{d}}), \quad s = 1, \dots, \frac{d}{2}$$



Более современный способ учёта относительных позиций:

$$c_j = \text{Attn}(q_j, K, V) = \sum_i (v_i + w_{i \boxplus j}^v) \text{SoftMax}_i a(k_i + w_{i \boxplus j}^k, q_j)$$

где $i \boxplus j = \max(\min(i - j, \delta), -\delta)$ — усечённая разность, $\delta = 5..16$

Vaswani et al. (Google) Attention is all you need. 2017.

Shaw, Uszkoreit, Vaswani. Self-attention with relative position representations. 2018.

Архитектура трансформера декодировщика

Авторегрессионный синтез последовательности:

$y_0 = \langle \text{BOS} \rangle$ — эмбединг символа начала;

для всех $t = 1, 2, \dots$:

1. Маскирование «данных из будущего»:

$$h_t = y_{t-1} + p_t; \quad H_t = (h_1, \dots, h_t)$$

2. Многомерное самовнимание:

$$h'_t = \text{LN} \circ \text{MH}_j \circ \text{Attn}(W_q^j h_t, W_k^j H_t, W_v^j H_t)$$

3. Многомерное внимание на кодировку Z :

$$h''_t = \text{LN} \circ \text{MH}_j \circ \text{Attn}(\tilde{W}_q^j h'_t, \tilde{W}_k^j Z, \tilde{W}_v^j Z)$$

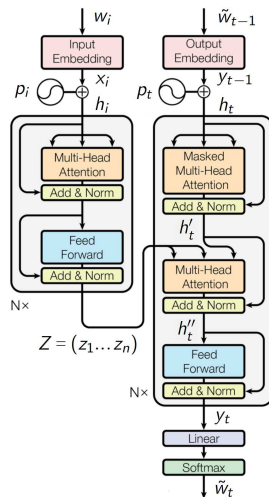
4. Двухслойная полносвязная сеть:

$$y_t = \text{LN} \circ \text{FFN}(h''_t)$$

5. Линейный предсказывающий слой:

$$p(\tilde{w}|t) = \text{SoftMax}_{\tilde{w}}(W_y y_t + b_y)$$

генерация $\tilde{w}_t = \arg \max_{\tilde{w}} p(\tilde{w}|t)$ пока $\tilde{w}_t \neq \langle \text{EOS} \rangle$



Vaswani et al. (Google) Attention is all you need. 2017.

Критерии обучения и валидации для машинного перевода

Критерий для обучения параметров нейронной сети W по обучающей выборке предложений S с переводом \tilde{S} :

$$\sum_{(S, \tilde{S})} \sum_{\tilde{w}_t \in \tilde{S}} \ln p(\tilde{w}_t | t, S, W) \rightarrow \max_W$$

Критерии оценивания моделей (недифференцируемые) по выборке пар предложений «перевод S , эталон S_0 »:

BiLingual Evaluation Understudy:

$$\text{BLEU} = \min\left(1, \frac{\sum \text{len}(S)}{\sum \text{len}(S_0)}\right) \text{mean}_{(S_0, S)} \left(\prod_{n=1}^4 \frac{\#n\text{-грамм из } S, \text{ входящих в } S_0}{\#n\text{-грамм в } S} \right)^{\frac{1}{4}}$$

Word Error Rate:

$$\text{WER} = \text{mean}_{(S_0, S)} \left(\frac{\#вставок + \#удалений + \#замен}{\text{len}(S)} \right)$$

Vaswani et al. (Google) Attention is all you need. 2017.

BERT (Bidirectional Encoder Representations from Transformers)

Трансформер BERT — это кодировщик без декодировщика, предобучаемый для решения широкого класса задач NLP

Схема преобразования данных в задачах NLP:

- $S = (w_1, \dots, w_n)$ — токены предложения входного текста
↓ обучение эмбедингов вместе с трансформером
- $X = (x_1, \dots, x_n)$ — эмбединги токенов входного предложения
↓ трансформер кодировщика
- $Z = (z_1, \dots, z_n)$ — трансформированные эмбединги
↓ дообучение на конкретную задачу
- Y — выходной текст / разметка / классификация и т.п.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova (Google AI Language)
BERT: pre-training of deep bidirectional transformers for language understanding. 2019.

Критерий MLM (masked language modeling) для обучения BERT

Критерий маскированного языкового моделирования MLM, строится автоматически по текстам (self-supervised learning):

$$\sum_S \sum_{i \in M(S)} \ln p(w_i | i, S, W) \rightarrow \max_W$$

где $M(S)$ — подмножество маскированных токенов из S ,

$$p(w | i, S, W) = \text{SoftMax}_{w \in V}(W_z z_i(S, W_T) + b_z)$$

— языковая модель, предсказывающая i -й токен предложения S ,
 $z_i(S, W_T)$ — контекстный эмбединг i -го токена предложения S
 на выходе Трансформера с параметрами W_T ,
 W — все параметры Трансформера и языковой модели

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova (Google AI Language)
 BERT: pre-training of deep bidirectional transformers for language understanding. 2019.

Критерий NSP (next sentence prediction) для обучения BERT

Критерий предсказания связи между предложениями NSP, строится автоматически по текстам (self-supervised learning):

$$\sum_{(S, S')} \ln p(y_{SS'} | S, S', W) \rightarrow \max_W,$$

где $y_{SS'}$ = [за S следует S'] — классификация пары предложений,

$$p(y|S, S', W) = \text{SoftMax}_{y \in \{0,1\}}(W_y \text{th}(W_s z_0(S, S', W_T) + b_s) + b_y)$$

— вероятностная модель бинарной классификации пар (S, S') ,
 $z_0(S, S', W_T)$ — контекстный эмбединг токена $\langle \text{CLS} \rangle$ для пары предложений, записанной в виде $\langle \text{CLS} \rangle S \langle \text{SEP} \rangle S' \langle \text{SEP} \rangle$

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova (Google AI Language)
 BERT: pre-training of deep bidirectional transformers for language understanding. 2019.

Ещё несколько замечаний про трансформеры

- **Fine-tuning:** для дообучения на задаче задаётся модель $f(Z(S, W_T), W_f)$, выборка $\{S\}$ и критерий $\mathcal{L}(S, f) \rightarrow \max$
- **Multi-task learning:** для дообучения на наборе задач $\{t\}$ задаются модели $f_t(Z(S, W_T), W_t)$, выборки $\{S\}_t$ и сумма критериев $\sum_t \lambda_t \sum_S \mathcal{L}_t(S, f_t) \rightarrow \max$
- *GLUE, SuperGLUE, Russian SuperGLUE* — наборы тестовых задач на понимание естественного языка
- Трансформеры обычно строятся не на словах, а на токенах, получаемых BPE (Byte-Pair Encoding) или WordPiece
- Первый трансформер: $N = 6$, $d = 512$, $J = 8$, весов 65M
- BERT_{BASE}, GPT1: $N = 12$, $d = 768$, $J = 12$, весов 110M
- BERT_{LARGE}: $N = 24$, $d = 1024$, $J = 16$, весов 340M

- Модели внимания сначала встраивались в RNN, но позже оказалось, что они самодостаточны
- Модель внимания работает точнее и быстрее RNN;
- легко предобучается и используется для многих задач;
- легко обобщается на тексты, графы, изображения
- Доказано, что модель внимания multi-head self-attention (MHSA) эквивалентна свёрточной сети [Cordonnier, 2020]
- Модель внимания лежит в основе Трансформера, различные варианты которого являются наиболее удачными моделями для понимания естественного языка: BERT, GPT-2/3/4, XLNet, ELECTRA и др.

Vaswani et al. Attention is all you need. 2017.

Dichao Hu. An Introductory Survey on Attention Mechanisms in NLP Problems. 2018.

Xipeng Qiu et al. Pre-trained models for natural language processing: A survey. 2020.

Cordonnier et al. On the relationship between self-attention and convolutional layers. 2020