Курс: управление научными проектами

Лекция 6

Лингвистический поворот в развитии компьютерных наук

30 октября
2024

ПОЛИТЕХ
Санкт-Петербургский
политехнический университет
Петра Великого

Содержание

Часть 1 (теория)

1. К истокам проблемы поиска объяснения
   - Энергетика процессов обучения и лингвистического «поворота» в развитии компьютерных наук

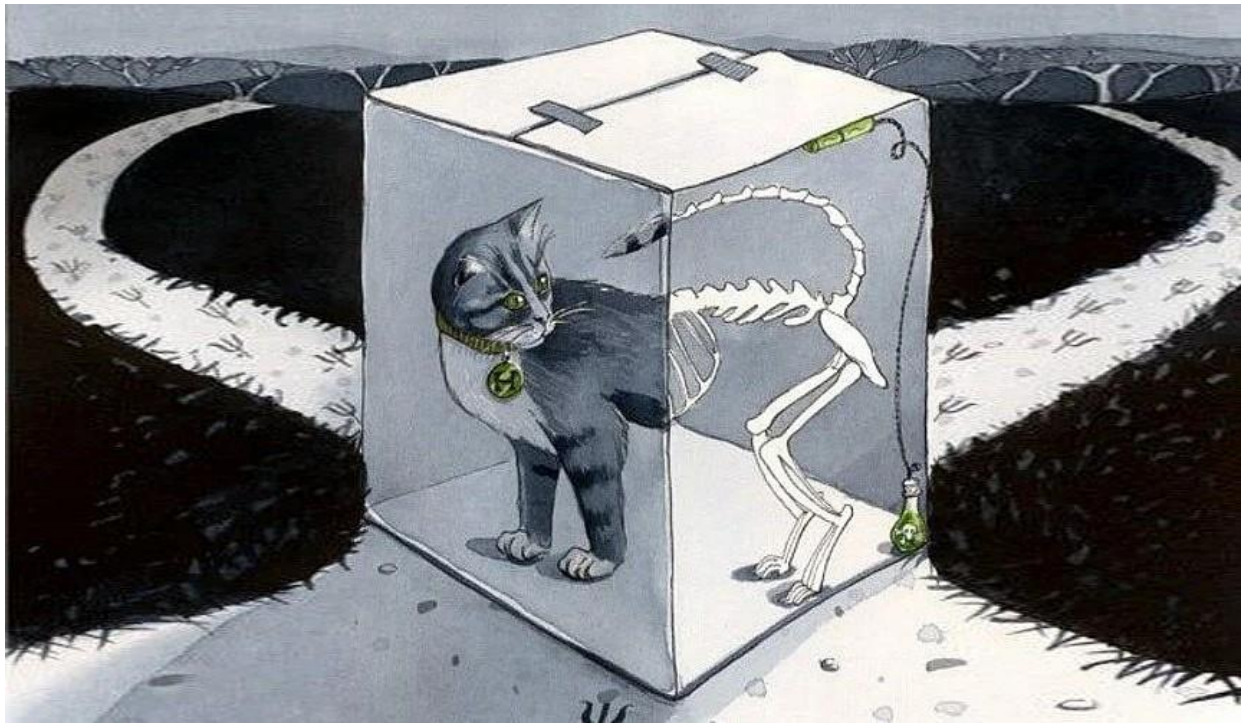
Часть 2 (практика)

3. Прикладные аспекты:

- От предобученных трансформеров к «машине Геделя»
- Концепция экзо-интеллекта: самоприменимость компьютерных технологий: от частично рекурсивных функций, реализуемых в МТ к самоприменимому трансформеру, генерирующему новые понятия

4. Выводы

Про кота Шредингера и «кота дяди Федора»



Эдуард Успенский

«Дядя Фёдор, пёс и кот» - у одних родителей мальчик был. Звали его дядя Фёдор:

- Здравствуй, мальчик. Ты зачем пришёл?

– Я хочу у вас про кота спросить.

– А что про кота?

– Допустим, у вас был кот ….

Согласно квантовой механике, если над частицей не производится наблюдение, то ее состояние описывается как квантовая суперпозиция (когерентная суперпозиция), т.е. **смешение всех возможных альтернативных состояний** в которых может находится частица.

***Итого: поведение квантовой частицы* не может быть предсказано никаким методом.**

Интеллектуальная суперпозиция: Физическое (наблюдаемое) vs информационное (возможное) - роль «кот Шредингера» в современной науке

Суть проблемы суперпозиции:
кот Шредингера наблюдает сам себя

Считается, что прямых аналогов суперпозиции квантово-механических состояний в доступных для **наблюдения макроскопических системах** не существует, поэтому такие системы всегда проявляют «объективные физические свойства», то есть их «свойства» детерминированы и ни как не зависят от того кто и как «наблюдает систему».
Прошлое в физическом смысле не может влиять на будущее…. только через «механизм памяти»

Очевидно, что если в описание интеллектуальной компьютерной системы включить влияние механизма памяти и учесть последствия «информационного воздействия» из прошлого, то …в проявленных системой свойствах появится фактор «наблюдателя» или «само-наблюдателя» , что будет влиять на…..ее будущие состояния .

Интеллектуальная система – это физическая система, которая имеет «прошлое» , то есть может «вернуться» в свое прошлое состояние

Как работают современные трансформеры: часть vs целое

- Долгое время для того, чтобы обрабатывать потоковые данные использовались Recurrent Neural Networks, RNN), Однако у RNN проблемы с обработкой длинных текстовых зависимостей: для полноценного анализа связного текста не достаточно перевода отдельных предложений, нужно учитывать общий контекст.
- В 2017 году был разработан «механизм внимания» (англ. attention), который позволяет при обработке (переводе) учитывать наиболее важные с точки зрения «смыла» части текста.
- С помощью механизма внимания нейросеть оценивает, какая позиция слов во входной последовательности важна для конкретной позиции последовательности слов на выходе.
- Семейство архитектур машинного обучения на основе механизма внимания получило название трансформеры (англ. transformers). Эта архитектура сочетает в себе:
- параллельную обработку данных,
- дообучения моделей
- применение механизма внимания.

Нейросеть-трансформер состоит из двух слоёв — энкодеров и декодеров

Энкодер — извлекает информацию из входящей последовательности (например, текста).

Декодер —использует извлечённую информацию для генерации элементов последовательности на выходе (например, текста на другом языке)жат несколько слоёв.

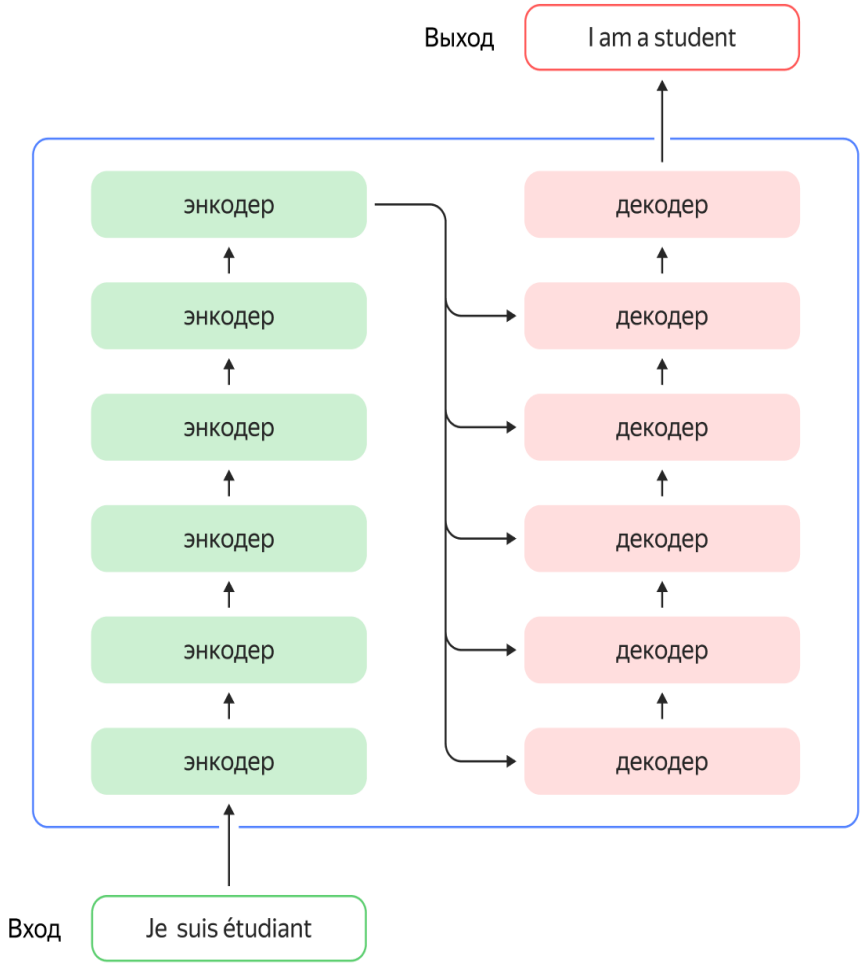
GPT (2018) -Generative Pre-trained Transformer

- GPT-2 генерировала результат перевода (обработки) входной последовательности без дополнительного обучения.
- GPT-2 могла делать краткое изложение и отвечать на вопросы исходя из содержания входного текста.
- GPT-3 была обучена на 500 миллиардах слов и использовала 175 миллиардов параметров, что позволяло ей качественно генерировать целые статьи и развёрнуто отвечать на вопросы, в том числе и достаточно узкоспециализированные

Как это работает: смысл не в слове, а в тексте из слов

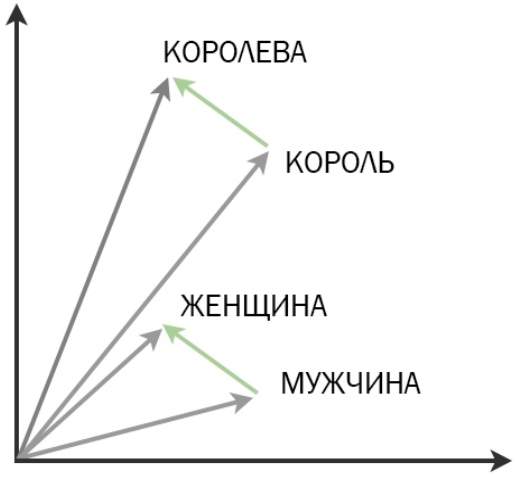
- Энкодер получает на вход набор токенов (части входной последовательности). Токеном может быть отдельное слово, знак пунктуации или частотная последовательность символов
- **Токены** конвертируются в последовательность **эмбеддингов** (цифровой формат\код), которые содержат информацию о **положении токена** во входной последовательности
- Эмбеддинги обрабатываются энкодером, что даёт возможность получать скрытые представления о контексте, в котором используется слово, что позволяет нейросети обрабатывать все части входных данных параллельно.
- На выходе из энкодера создается модель входной последовательности как набор векторов — скрытых представлений входных данных. Эти вектора передаются декодеру, который, **используя механизм внимания, распаковывает его в целевую последовательность**.
- Это может быть предложение, что было на входе, но записанное на другом языке.

схема



обучить трансформер можно на большом наборе данных - датасете в режиме языковой модели, а затем дообучать модель на малых данных для конкретных задач, то результат оказывается существенно лучше, чем раньше. Именно так реализовывались проекты BERT (Bidirectional Encoder Representations from Transformers) и GPT (Generative Pre-trained Transformer).

Механизм Word2Vec: слово переводится в число и контекст



Технология Word2Vec работает с текстовым корпусом и по определенным правилам присваивает каждому слову уникальный набор чисел — семантический вектор.
Если из вектора слова *«король»* вычесть вектор *«мужчина»* и прибавить вектор *«женщина»*, получатся вектор , соответствующие слову *«королева»*.

Можно ли смысл слова выразить числом ? Нет. Семантический вектор показывает, как часто данное слово встречалось рядом с другими словами.

Дистрибутивная гипотеза. смысл слова заключается не в наборе его собственных букв, а в том, среди каких слов слово чаще всего встречается. То есть смысл слова распределен между элементами его возможных контекстов
с этой точки зрения слово «котёнок» окажется сильнее связан по смыслу со словом *«щенк»*, потому что оба они встречаются рядом со словами *милый, пушистый, маленький*. В то же время со словом *«стол» слово «котёнок»* почти не связан, так как котята редко бывают «ровными», «пластиковыми» и »деревянными».

Детали «обучения» и построения семантических векторов

- Для описания слов с помощью чисел используется квадратная таблицу, в которой каждая строчка — это слово из словаря большого текстового корпуса. столбцы таблицы — это те же самые слова. В ячейке на пересечении столбца и строки пишется число раз, которое слово из строки встретилось в корпусе рядом со словом из столбца.
- Семантические вектора - это строки нашей таблицы. Используется специальная мера схожести слов, представленных семантическим вектором известная как «косинусная близость».
- Если семантические векторы двух слов «косинусно близки» по отношению друг к другу, то эти векторы принадлежат словам, близким по смыслу в человеческом понимании.
- Если слово в корпусе находится «рядом» это значит, что оно находится на расстоянии не более N слов. Это расстояние называются «шириной окна» поиска. Обычно принято использовать N = 10. В итоге, при обработке входной текст разбивается на отрывки из подряд идущих N слов - N-граммы,

N-грамма

N-грамма — это последовательность из n элементов (слогов, слов или символов), идущих подряд в анализируемом тексте\файле. Последовательность из двух элементов называют **биграмма**, из трёх элементов — **триграмма**.

Вычислив частоту вхождения N-грамм в текстах корпуса, можно оценит скрытые паттерны текстов.

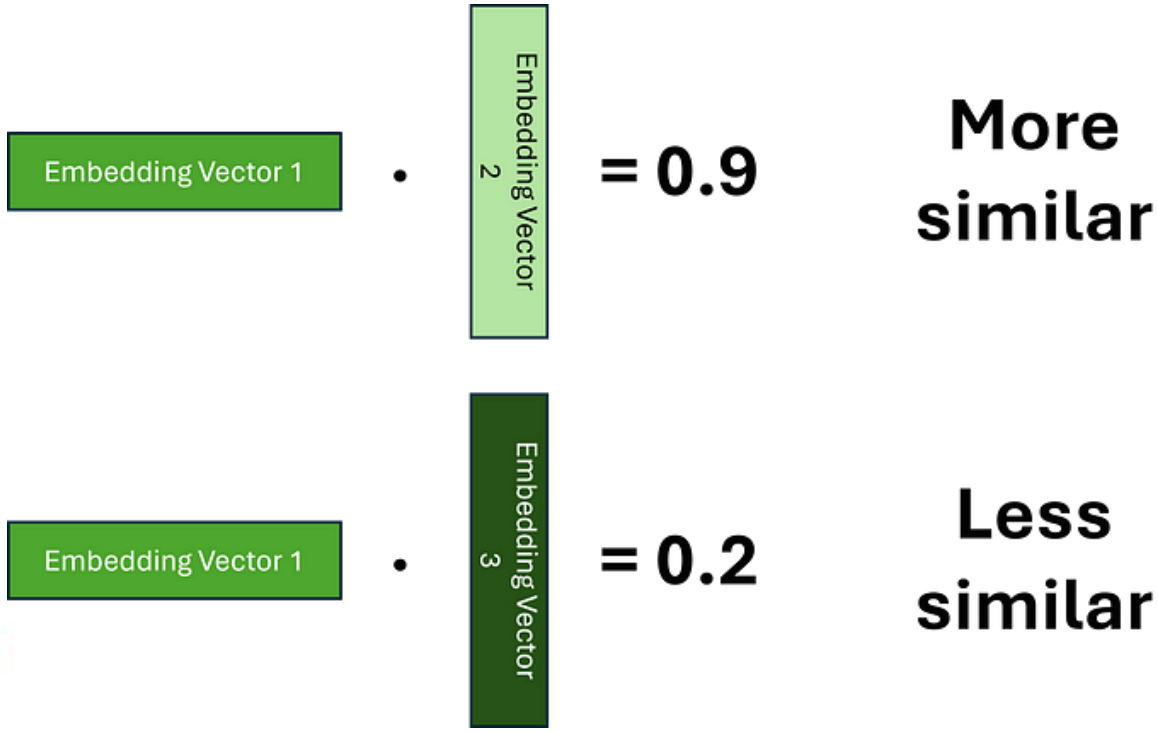
N-граммы используются в следующих задачах:
•Выдачи подсказок следующего слова (например, в поисковой строке). N-граммная модель позволяют вычислить вероятность следующего слова N-граммы, если известны предыдущие.
•Выявления авторства или плагиата. Можно вычислить N-граммы для разных текстов и сравнить степень сходства.
•Поиска и коррекции ошибок

•при сложении векторов двух слов получается вектор третьего слова, и оно «суммирует» смыслы двух слагаемых в человеческом понимании, иногда неожиданно или даже поэтично.

Как вычисляется внимание?

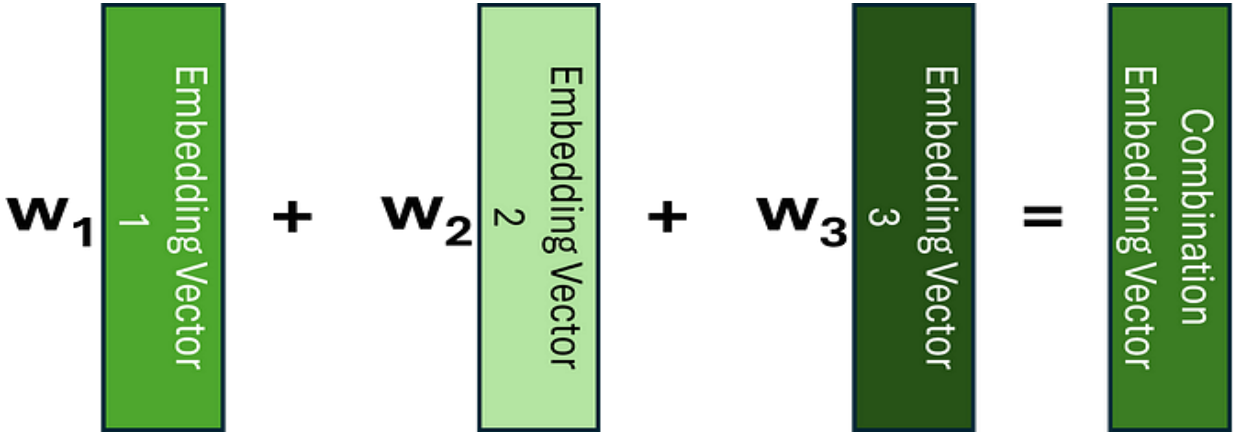
- Есть разные варианты этого механизма, и главное различие между ними заключается в **способе вычисления весов для линейной комбинации**. Базовый механизм реализован основе скалярного произведения .
- Будем считать, что все эмбеддинги позиционно закодированы.
- Цель «внимания» — создать контекстуализированные эмбеддинги с помщью линейных комбинаций исходных эмбеддингов. Для этого нужно определить, какие токены релевантны друг другу. Понятие сходства между двумя эмбеддингами можно задать через скалярное произведение. Чем больше это произведение, тем больше сходство двух слов.
-

Чем больше произведение, тем больше сходство двух слов ?!.



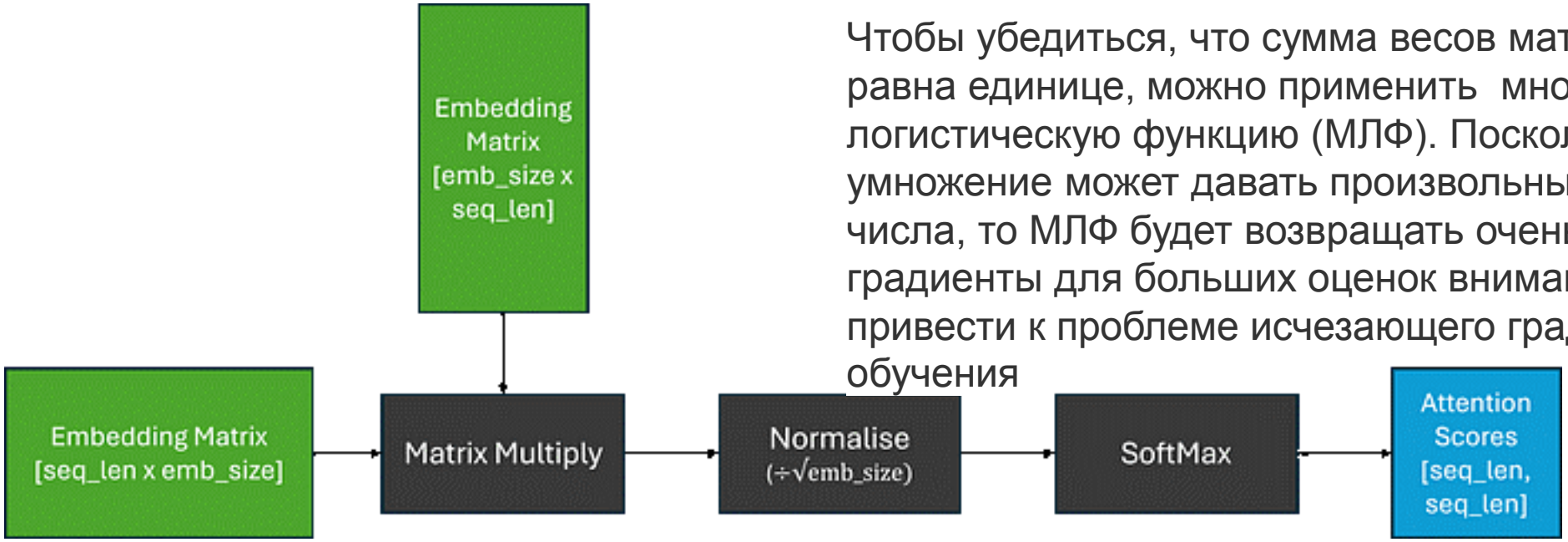
Поскольку для каждого токена нужно вычислить его взаимосвязь с каждым другим токеном последовательности, мы можем обобщить задачу до матричного умножения и получим матрицу весов, которую часто называют *attention scores (оценка внимания)*.

Линейную комбинацию эмбеддингов обычно представляют так:



- Заметим, что до применения механизма внимания у эмбеддингов слов входной последовательности нет контекста по отношению к соседним с ними слов в предложнении. Применив механизм внимания, мы можем вычислить матрицу весов $\{w_i\}$, которая позволит выразить комбинированный вектор эмбеддинга для всех слов обрабатываемой входной последовательности.
- Эмбеддингам, соответствующим наиболее релевантным для выбранного токена частям последовательности, должны быть присвоены веса большего размера (величины). Это должно обеспечить количественное выделение в новом **комбинированном векторе эмбеддинга** самого «важного» контекста. Эмбеддинги, содержащие информацию об их текущем контексте называют *контекстуализированными*, и именно такие эмбеддинги формирует нейросеть с архитектурой трансформера.

итак

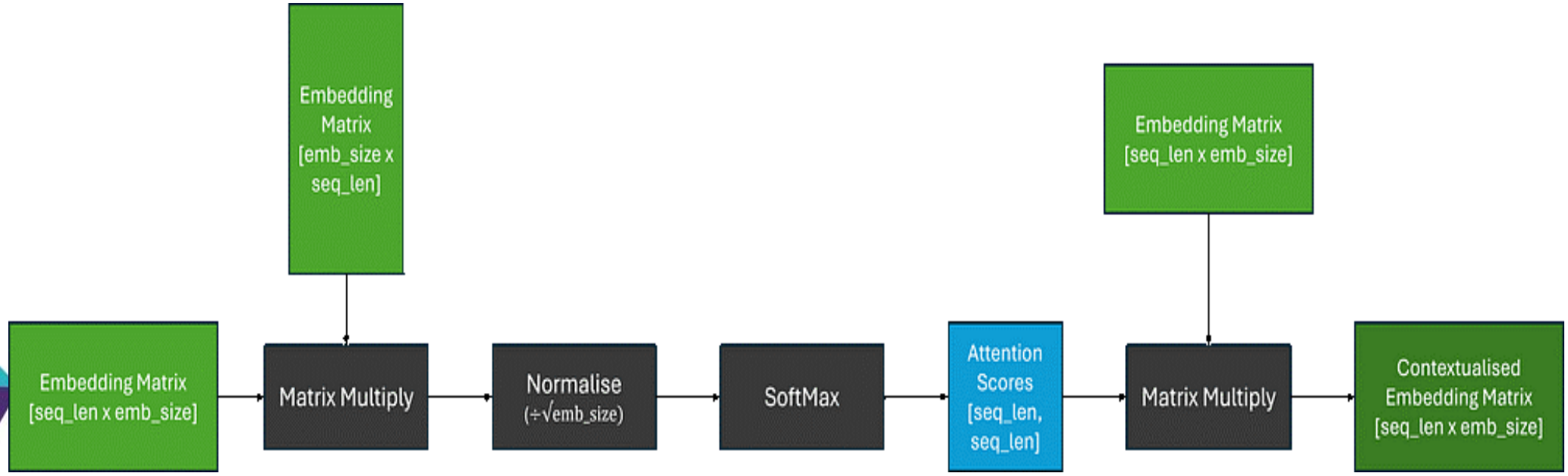


Чтобы убедиться, что сумма весов матрицы внимания равна единице, можно применить многопеременную логистическую функцию (МЛФ). Поскольку матричное умножение может давать произвольные большие числа, то МЛФ будет возвращать очень маленькие градиенты для больших оценок внимания, что может привести к проблеме исчезающего градиента во время обучения

для получения матрицы контекстуализированных эмбеддингов можно умножить оценки внимания на матрицу исходных эмбеддингов. Это равнозначно взятию линейных комбинаций наших эмбеддингов.

Упрощенное вычисление внимания. .

Предполагается, что эмбеддинги позиционно закодированы



Масштабированное вычисление внимания (self-attention) на основе скалярного произведения.

для облегчения обучения модели вместо прямого использования матрицы эмбеддингов будем использовать три независимых линейных слоя (матричных умножений).

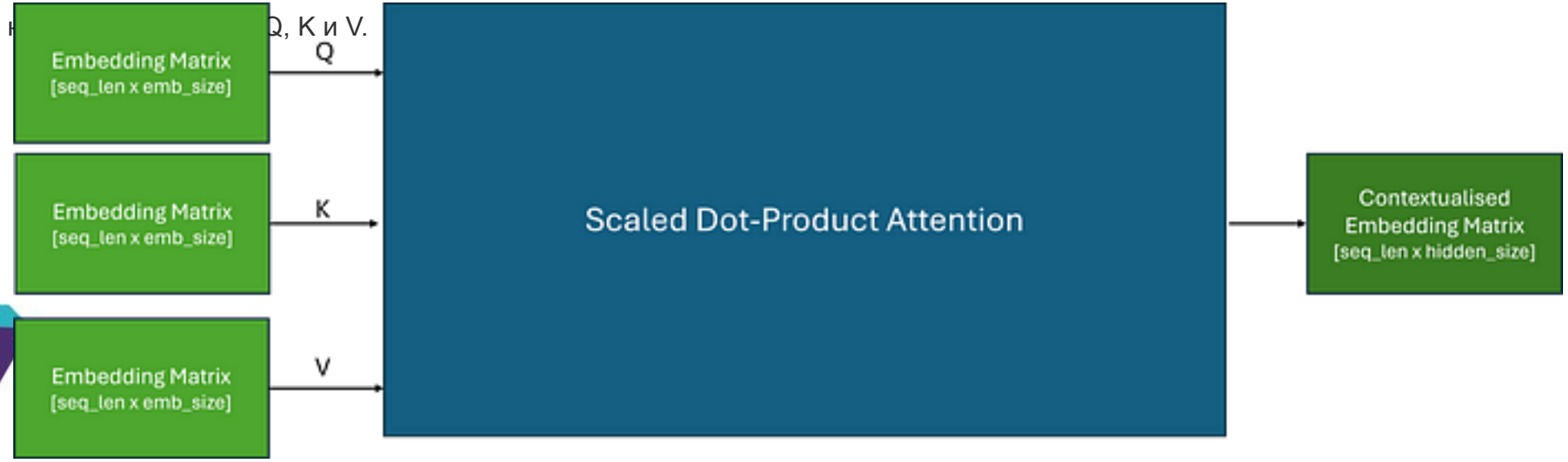
линейные проекции помечены как Q, K и V. В исходной научной работе они названы *Query, Key* и *Value*, вероятно, авторы вдохновлялись процессом извлечения информации.

ПОЛИТЕХ

МЫСЛИТЬ БУДУЩИМ

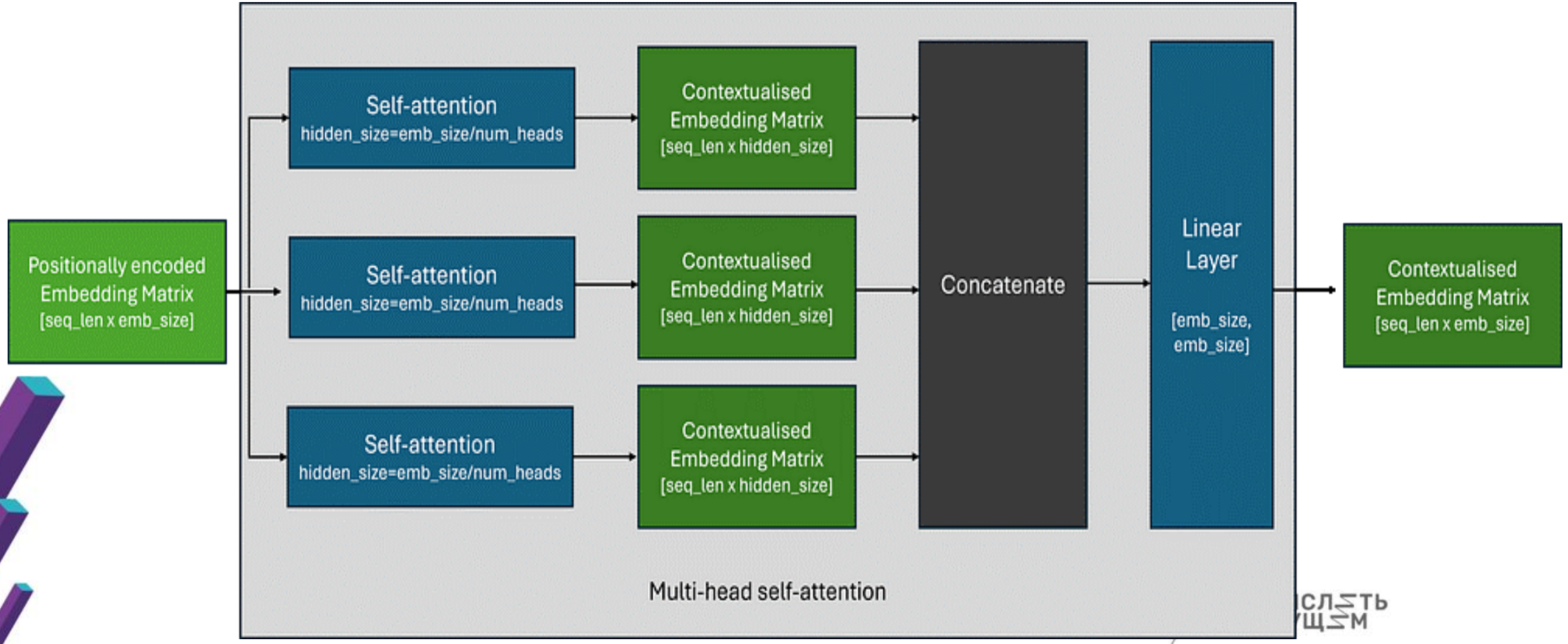
Вычисление внимания можно представить как блок с тремя входами, которые подаются в Q, K и V.

Когда подаем одну матрицу в Q, K и V, это называется *self-attention* — выявление закономерностей только между входными данными.



Что такое multi-head attention?

На практике мы часто параллельно запускаем несколько блоков self-attention, чтобы трансформер одновременно обрабатывал разные части входной последовательности — это называют *multi-head attention*. Идея проста: выходы нескольких независимых блоков self-attention конкатенируются и передаются через линейный слой. Он позволяет модели комбинировать контекстуальную информацию из каждого блока внимания.



Что ещё входит в состав трансформера?

- название работы, в которой впервые был описан трансформер «*Вам нужно лишь внимание*» (*Attention is all you need*) вводит в заблуждение, потому что в трансформере есть много других компонентов:
- **Нейросеть с прямой связью** (**Feedforward Neural Network**, FFN): двухслойная нейросеть, которая применяется независимо к каждому эмбеддингу токена в пакете и последовательности. Её задача — внести в трансформер дополнительные изучаемые параметры, позволяющие убедиться, что контекстуальные эмбеддинги разделены и распределены.
- **Нормализация слоев** (Layer Normalization): помогает стабилизировать обучение нейросетей, в том числе трансформеров. Он нормализует активации для каждой последовательности, не позволяя им стать слишком большими или маленькими во время обучения, что может приводить к проблемам вроде исчезающего или взрывающегося градиента.
- **Связи с пропускам** (**Skip connections**): как и в архитектуре ResNet, для борьбы с проблемой исчезающего градиента и повышения стабильности обучения используются остаточные подключения.
-

Какие бывают трансформеры?

- Существует много разных трансформерных архитектур, и большинство можно разделить на три типа.
  - **Энкодеры.** Модели-энкодеры синтезируют контекстуальные эмбеддинги, которые можно использовать в последующих задачах вроде классификации или распознавания именованных сущностей, поскольку механизм внимания может обрабатывать всю входящую последовательность.. Самое популярное семейство чистых трансформеров-энкодеров — это BERT и его разновидности.
- передав данные через один или несколько блоков-трансформеров, мы получаем сложную матрицу контекстуализированных эмбеддингов, которая содержит по эмбеддингу на каждый токен последовательности. Но чтобы использовать эти данные для последующих задач вроде классификации нужно сделать одно предсказание. Обычно берут первый токен и передают через классификатор, в котором есть слои Dropout и Linear.
- Результат работы этих слоев можно пропустить через МЛФ для превращения в вероятности классов.

Еще один вид - **декодеры**

- Этот тип архитектур почти идентичен предыдущему, **главное отличие в том, что декодеры используют *маскированный (или причинный) слой self-attention***, поэтому механизм внимания может принимать только текущий и предыдущие элементы входной последовательности.
- То есть контекстуальные эмбеддинги учитывают только предыдущий контекст. К популярным моделям-декодерам относится семейство GPT.

Многокомпонентная архитектура «Энкодеры-декодеры»

- Изначально трансформеры были представлены **как архитектура для машинного перевода и использовали и энкодеры, и декодеры**. С помощью энкодеров создается промежуточное представление, прежде чем с помощью декодера переводить в желаемый формат.
- Хотя энкодеры-декодеры сегодня менее распространены, архитектуры вроде T5 показывают, что задачи вроде ответов на вопросы, подведения итогов и классификации можно представить в виде преобразование последовательности в последовательность и решить с помощью описанного подхода.
- Главное отличие архитектур типа энкодер-декодер заключается в том, что декодер использует *энкодер-декодерное внимание*: при вычислении внимания используется результат энкодера (K и V) и входные данные декодера (Q). Сравните с self-attention, когда для всех входных данных используется одна и та же входная матрица эмбеддингов. При этом общий процесс синтеза очень похож на процесс в архитектурах декодеров.

Итак: Механизм внимания: ключевая особенность трансформеров

- После того как входной текст был преобразован в последовательность векторов, эта последовательность поступает в центральный механизм трансформера – **блок внимания** (attention block).
- Это одна из ключевых особенностей, которая отличает трансформеры от предыдущих архитектур и во многом определяет их выдающиеся способности.
- Идея внимания заключается в том, что векторы, представляющие отдельные фрагменты текста, должны иметь возможность «общаться» друг с другом и **обмениваться информацией**. Это позволяет им обновлять свои значения с учетом контекста, в котором они встречаются.

Итого: архитектуру трансформера

Собрав все компоненты рассматриваемой системы воедино, а именно блоки :

- разбиение на токены,
- создание векторных представлений,
- механизм внимания,
- многослойные персептроны
- финальное предсказание, –

мы получаем **архитектуру трансформера**, способную манипулировать текстами и генерировать осмысленные ответы

Как формально можно представить «механизм внимания»

Дана непараметрическая модель «черного ящика
«вход» {vi}» -значение выход» (y) :

$$y = \sum_{i=1}^{N} \alpha(q, k_i) v_i$$

$$\alpha(q, k_i) = softmax_i\left(score(q, k_j)\right)_{j=1}^{N}$$

q – query - запрос
k_i – key - ключ
v_i – value – значение

«механизм внимание» для такого описания процесса машинного обучения это преобразование «ядра» непараметрической модели $\alpha(q,k\_i)$ в оценку «вероятности » некоторого действия (подстановки в генерируемое предложение «нужного» слова)

- Трансформенное преобразование можно использовать для обработки сложных последовательностей, например предложений естественного языка
- Такое преобразование позволяет «работать» с произвольным числом пар векторов-эмбеддингов (ключ, значение)
- Обычно функция $score(q, k_i) \propto q^T k_i$, либо линейная функция от q и k, либо матричный оператор вида $\left(W_Q q\right)^T (W_K k_i) = q^T W_Q^T W_K k_i = q^T W k_i$

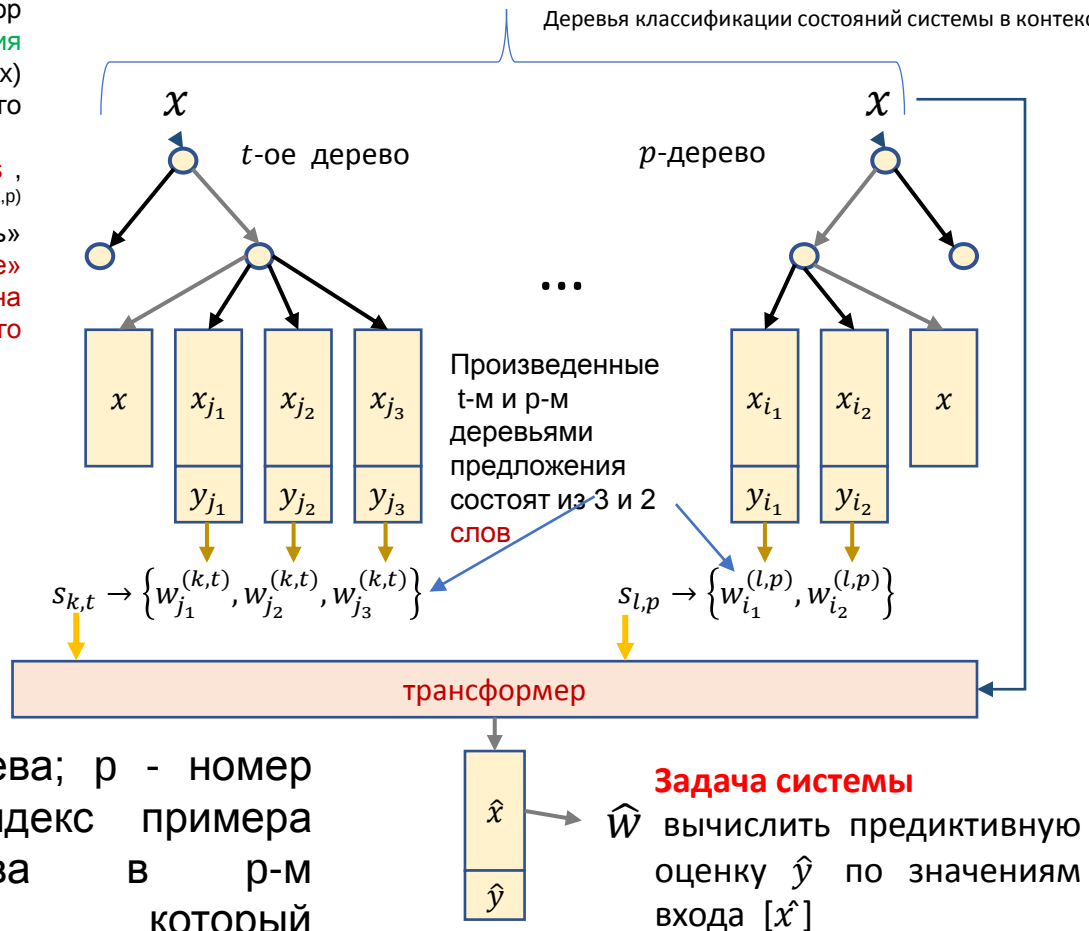
Многокомпонентная архитектура классификации результатов мониторинга объекта и их трансформации в оценки состояния объекта

Имеется система описание которой задается текстом. Для системы сформирована **обучающая выборка** (x_j,y_j), где x – вектор признаков (давление, частота..), y - реакция организма/системы и индексы $j \in J_k(x)$ признаков x в один и тот же p-й лист k-го дерева классификации.
(xj,yj), представим в виде предложения s , состоящего из слов, обозначаемых как $w_i^{(k,p)} \in J_k(x)$, из которых можно «сформировать» актуальное «лингвистическое описание» текущего состояния в контексте (на основании) значения текущего входного сигнала x

$i \in J_k(x)$,

$s_{k,p} = \{w_{i1}^{(k,p)}, \ldots, w_{ip}^{(k,p)}\}$

«предложение» s:



ИЗ КАЖДОГО «ЛИСТА» ДИРЕВЬЕВ МОЖЕТ ВЫХОДИТЬ РАЗНОЕ КОЛИЧЕСТВО «ВЕТВЕЙ»:

1) из листа t-ого дерева выходит три «слова» описания реакции системы , а из листа p-дерева выходит описание, состоящее из двух «слов»
2) На вход системы подается текущий вектор "x" и соответствующие ему значения выхода системы

k - номер дерева; p - номер листа; i - индекс примера (индекс слова в p-м предложении), который попадает в p-й лист.

Задача системы

$\widehat{W}$ вычислить предиктивную оценку $\hat{y}$ по значениям входа $[\hat{x}]$