1 Введение

В последние годы, методы машинного обучения позволяют добиваться успехов в задачах рекомендации, однако задачи требующие подбора группы объектов совместимых между собой все еще представляют серьезную проблему. Основным препятствием является неизвестный характер зависимости между объектами в таких группах. В работе исследуется необходимость и способы построения аппроксимации этой зависимости для прикладной задачи рекомендации наборов одежды называемых «образами». В частности, в отличие от ранее представленных работ, рассматриваются методы решения задачи рекомендации более одного элемента одежды, дополняющих заданный «образ». Предлагается рассмотреть двухшаговый подход к приближенному решению дискретной оптимизационной задачи, превосходящий по скорости ранее представленные и сравнить его качество с методом жадного перебора – лучшим вычислимым за ограниченное время. Для аппроксимации зависимости между элементами образа предлагается использовать графовые нейронные сети (GNN) [13].

...

2 Постановка задачи

2.1 Основные понятия

Основная единица данных, рассматривающаяся в работе – элемент одежды, далее будем называть его *объектом* или *элементом*, множество всех рассматриваемых объектов – $\mathcal{X}$.

Каждый объект $X \in \mathcal{X}$ представим как пару $X = (I, T)$ соответственно изображения объекта (может отсутствовать) и его текстового описания(может быть пустым). Пусть $\mathcal{I}$ – множество изображений объектов.

Некоторое подмножество $O = \{X_i\}_{i=1}^{k} \subset \mathcal{X}$ множества всех элементов будем называть *образом*, если

1. $O \neq \{\emptyset\}$
2. $|O| \leqslant K$

где K – определяемая задачей константа. Множество всех образов обозначим $\mathcal{O}$.

Для элементов и образов будем рассматривать *функции близости*

$$S_X : \ \mathcal{X} \times \mathcal{X} \longrightarrow [-1, 1], \ \ \forall X \in \mathcal{X} \ S_X(X, X) = 1$$

$$S_O : \ \mathcal{O} \times \mathcal{O} \longrightarrow [-1, 1], \ \ \forall O \in \mathcal{O} \ S_O(O, O) = 1$$

Такой функцией может выступать например косинусное сходство в некотором латентном пространстве.

Для оценки образов введем функцию *оценки* или *совместимости* его элементов:

$$\mathcal{S} : \ 2^{\mathcal{X}} \longrightarrow [0, 1]$$

причем выполнено следующее:

$$\forall O \in \mathcal{O} : \ \mathcal{S}(O) > 0$$

$$\forall O' \in 2^{\mathcal{X}} \setminus \mathcal{O} : \mathcal{S}(O') = 0$$

Совместимостью или *оценкой совместимости* образа O будем называть результат применения функции совместимости к этому образу $\mathcal{S}(O)$

2.2 Описание рассматриваемых задач

2.2.1 Оценка образа

Выше введена функция $\mathcal{S}$, ассоциирующая с каждым образом его оценку совместимости, однако вид такой функции неизвестен. Задача оценки образа – это классическая задача регрессии:

- **Дано:**
  $\{O_1 \dots O_n\} \subset \mathcal{O}$
  $\{\mathcal{S}(O_1) \dots \mathcal{S}(O_n)\}$
- **Требуется:**
  Найти наилучшую в некотором смысле аппроксимацию функции $\mathcal{S}$ функциями заданного класса, т.е. решить задачу оптимизации:

$$\hat{\mathcal{S}} = \underset{S}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\mathcal{S}(O_i), S(O_i)) \right]$$

  где $\mathcal{L}(\cdot, \cdot)$ некоторая метрика, например евклидова

2.2.2 Описание образа (скорее всего уберу)

Задача описания образа есть задача построения наилучшего текстового описания для данного образа по изображениям его элементов. Полагая функцию оценки известной получаем следующую формальную постановку:

- **Дано:**
  $\{X_i\} = O_n,\ |O_n| = n$ — образ, где $X_i = (I_i, \emptyset),\ I_i \in \mathcal{I},\ i = \overline{1, n}$ — его элементы с пустым текстовым описанием.
  $O_n(T) = \{X_i(T) = (I_i, T)\}_{i=1}^{n}$ для некоторого общего описания T

- **Требуется:**
  Найти оценку $\hat{T}$ общего для всех элементов описания T, максимизирующую значение функции оценки $\mathcal{S}(O_n)$, т.е.:

$$\hat{T} = \underset{T}{\operatorname{argmax}}\, \mathcal{S}(O_n(T))$$

Стоит заметить, что в данном случае задача генеративная — лучшее описание находится, из решения задачи максимизации функции оценки, а не просто выбирается из заранее заданного конечного множества.

2.2.3 Дополнение (восстановление) образа

Для решения задачи восстановления образа необходимо знать функцию совместимости или ее аппроксимацию. Предполагая ее известной, получаем следующую постановку:

- **Дано:**
  $O_n \in \mathcal{O},\ |O| = n$
  $k \in \mathbb{N},\ k > n$ — количество недостающих элементов
  $\{\hat{T}_i\}_{i=1}^{k}$ — текстовые представления недостающих элементов, возможно пустые. В случае если предлагается только текстовое описание всего образа T, оно используется в качестве описания каждого элемента, т.е. $\forall i \in \overline{1, k} :\ T_i = T$.

- **Требуется:**
  Найти наилучшее в смысле максимизации функции оценки дополнение образа O_n до $O_k \in \mathcal{O}, |O_k| = k$, т.е. решить следующую задачу:

$$\{\hat{X}_i\}_{i=1}^k = \underset{\{X_i\}_{i=1}^k \in \mathcal{X}^k}{\operatorname{argmax}} \left[\alpha \cdot \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right) + (1-\alpha) \cdot \sum_{i=1}^{k} S_X((I_i, T_i), (I_i, \hat{T}_i)) \cdot \mathbb{I}\{\hat{T}_i \neq \emptyset\} \right]$$

  здесь $X_i = (I_i, T_i)$, $\alpha \in [0, 1]$. Второе слагаемое отвечает за соответствие предсказанного элемента предъявленному текстовому представлению и равно нулю, если представление пусто. Задача, в отличие от предыдущей, дискриминативная и может быть решена точно полным перебором всех объектов из $\mathcal{X}$.

2.2.4 Генерация образа (скорее всего уберу)

Задача генерации состоит в выборе образа произвольного размера, наиболее подходящего к предоставленному текстовому описанию. В терминах введенных выше получаем:

- **Дано:**
  T — текстовое описание образа.

- **Требуется:**
  Найти наилучший в смысле максимизации функции оценки образ $O \in \mathcal{O}$ элементы которого наилучшим образом соответствуют предложенному описанию T, т.е.:

$$\hat{O} = \underset{k, O=\{X_i\}_{i=1}^k, O \in \mathcal{O}}{\operatorname{argmax}} \left[\alpha \cdot \mathcal{S}(O) + (1-\alpha) \cdot \sum_{i=1}^{k} S_X((I_i, T_i), (I_i, T)) \cdot \mathbb{I}\{\hat{T} \neq \emptyset\} \right]$$

  здесь $X_i = (I_i, T_i)$, $\alpha \in [0, 1]$. Стоит заметить, что если зафиксировать $k = 1$, получаем обычную задачу поиска наиболее подходящего под описание элемента в коллекции. Учитывая то, что множество образов определено множеством элементов и определением образа, задача генерации образа также является дискриминативной и состоит в переборе всех возможных образов.

2.3 Обзор литературы

2.3.1 Подходы к построению функций близости

При рассмотрении проблемы построения достаточно информативной функции от образов встает вопрос о том какие особенности структуры исходных данных могут быть полезны и на чем можно основывать выбор архитектуры.

Поскольку образ состоит из переменного числа первоначально независимых элементов, естественно перед рассмотрением образа независимо получать некоторое представление каждого его элемента. Кроме того, понятно, что операции с элементами внутри образа должны быть эквивариантны к перестановке элементов, поскольку на них не возникает естественного порядка. Развивая эту идею, можно сказать что каждому элементу присущ некоторый набор признаков, часть из которых может существенно влиять на совместимость, а значит декомпозиция элементов на признаки, моделирование их взаимосвязей а также взаимосвязей элементов между собой может помочь в построении оценки всего образа.

Подобным образом декомпозированная на разные масштабы задача может быть представлена например в виде графа, а значит одним из способов решения будут графовые нейронные сети. Такой подход в контексте оценки и подбора образов рассматривают в частности в статьях «Fashion Retrieval via Graph Reasoning Networks on a Similarity Pyramid» [9] и «Hierarchical

Fashion Graph Network for Personalized Outfit Recommendation» [10].

В [9] авторы предлагают GRNet – модель для определения похожести элементов, которая обрабатывает несколько представлений изображения каждого элемента (текстовые описания в статье не рассматривается) в разном масштабе. Т.е. авторы использует еще один промежуточный уровень – масштаб – для построение латентного представления каждого элемента.

В предложенной модели для каждого элемента сначала с помощью GoogLeNet [15] создаются несколько представлений как раз и называемых масштабами, далее несколько вырезанных частей каждого из полученных представлений называются признаками. Считаются локальные векторы близости между соответствующими признаками в соответствующих масштабах, в конце осуществляется message passing [3] по полному графу, вершинами которого являются вычисленные векторы. Таким образом, вместо рассмотрения единого векторного представления, каждый элемент, пользуясь предполагаемой симметрией задачи, декомпозируют на несколько масштабов, а каждый масштаб на несколько признаков и моделируют их взаимодействие.

Оставим за пределами нашего рассмотрения подробности процесса получения признаков, тогда пусть для элементов $X, Y \in \mathcal{X}$, $\{x_l^i \in \mathbb{R}^{C\times 1}\}$ и $\{y_l^i \in \mathbb{R}^{C\times 1}\}$ векторные представления i-го локального признака в l масштабе. Для каждого масштаба l и номеров признаков i и j вектор локальной близости — s_l^{ij}:

$$s_l^{ij} = \frac{P|x_l^i - y_l^j|^2}{\|P|x_l^i - y_l^j|^2\|_2}$$

где P – так называемая проекционная матрица с обучаемыми весами.
Из этих векторов составляется граф, называемый авторами пирамидой. Далее для каждой пары вершин $s_{l_1}^{ij}$ и $s_{l_2}^{mn}$ определяется скаляный вес $w_p^{l_1 ijl_2 mn}$:

$$w_p^{l_1 ijl_2 mn} = \frac{\exp((\mathbf{T}_{out} s_{l_1}^{ij})^\intercal (\mathbf{T}_{in} s_{l_2}^{mn}))}{\sum\limits_{l,p,q} \exp((\mathbf{T}_{out} s_{l_1}^{ij})^\intercal (\mathbf{T}_{in} s_l^{pq}))}$$

где $\mathbf{T}_{in}, \mathbf{T}_{out}$ – обучаемые матрицы. Тогда для $l_1 = l_2$ это веса внутри одного масштаба, а для $l_1 \neq l_2$ – между разными, что позволяет им "сообщаться". Таким образом мы определили граф близости $G = (\mathbb{N}, \mathbb{E})$, где $\mathbb{N} = \{s_l^{ij}\}$ а $\mathbb{E} = \{w_p^{l_1 ijl_2 mn}\}$.
Далее вектора в каждой вершине обновляются по следующему правилу:

$$\widehat{s}_{l_1}^{ij} = ReLU\left(\mathbf{W} \sum_{l_2,m,n} w_p^{l_1 ijl_2 mn} s_{l_2}^{mn}\right)$$

где $\mathbf{W}$ – еще одна обучаемая матрица параметров. Итоговая оценка близости получается в дополнительной вершине графа, которая предварительно инициализируется некоторой простой функцией s_g:

$$s_g = S_g(A(X), A(Y))$$

где $A(.)$ — например average pooling или max pooling по всем признакам во всех масштабах, а S_g — например косинусное сходство.

В [10] авторы используют схожий подход к декомпозиции для задачи рекомендации образов. Рассматривается иерархическая структура из трех уровней — уровня элементов, уровня образов уровня пользователей. Для рассматриваемых пользователей, образов и элементов всех этих образов строится соответствующих трехуровневый граф с ребрами направленными с уровня элементов в уровень образов и из уровня образов в уровень пользователей. Далее, начиная с нижнего уровня, с помощью фактически чуть упрощенного подхода графовых сверточных сетей [8], латентные представления элементов последовательно агрегируются в латентные представления на уровне образов и на уровне пользователей, которые непосредственно используются

для рекомендации путем выбора для пользователя с представлением u^* образа $\hat{O}$ по правилу

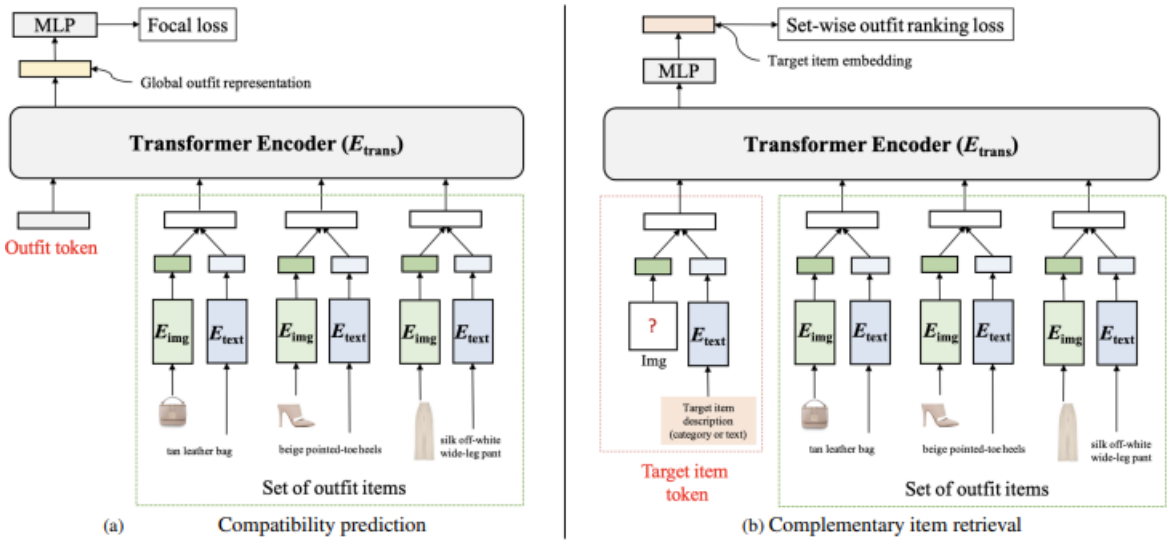
$$\hat{O} = \underset{O \in \mathcal{O}}{\operatorname{argmax}} f(O)^\intercal u^*$$

где $f(O)$ — представление образа, полученное при подсчете в том же графе.

2.3.2 Восстановление, генерация и оценка образов

В задачах оценки, генерации и рекомендации образов, как и во многих других, хорошо показывают себя специальным образом обученные трансформерные [16] архитектуры. Заметим, что в силу отсутствия порядка на элементах и их признаках трансформер без позиционного кодирования [16] фактически эквивалентен graph attention network [17] с несколькими LayerNorm [1] и линейными слоями. Таким образом, при использовании этой архитектуры мы также неявно учитываем рассмотренную выше симметрию данных. Варианты применение трансформеров для задач связанных с образами рассмотрены в частности в статьях [5] и [12].

В [12] авторы предлагают использовать архитектуру трансформера для обучения представления образа целиком. Полученное представление используется для оценки совместимости образа и восстановления образа с пропущенными элементами. В качестве токенов в трансформер подаются латентные представления элементов, т.е. фактически обрабатывается полный граф из всех элементов образа. Помимо изображений элементов, авторы также используют их текстовое описание и одновременно тренируют 2 энкодера для текстового и графического представления каждого объекта. Для задачи восстановления/рекомендации в модель также подается дополнительный токен, инициализированный случайный образом, после прохождения через модель агрегирующий в себе информацию из всех остальных и использующийся в итоге непосредственно для выбора ближайшего в смысле максимизации скалярного произведения дополняющего образ элемента.



(перерисовать картинку из статьи??)

Предложенную модель обучают сначала для оценки совместимости образа, а затем заменяют последние линейные слои и дообучают получать представление образа для подбора подходящих элементов на основании полученного представления и текстового описания нового элемента. Из этого можно заключить, что учет внутренней структуры оказывается полезным и для оценки образа целиком, и для восстановления отдельных его частей, причем для разных задач возможно использовать одни и те же латентные представления элементов.

Идею с общими внутренними представлениями для разных задач развивают авторы [5]. В статье рассматривается вопрос обучения мультимодального трансформера одновременно на

несколько задач связанных с образами.

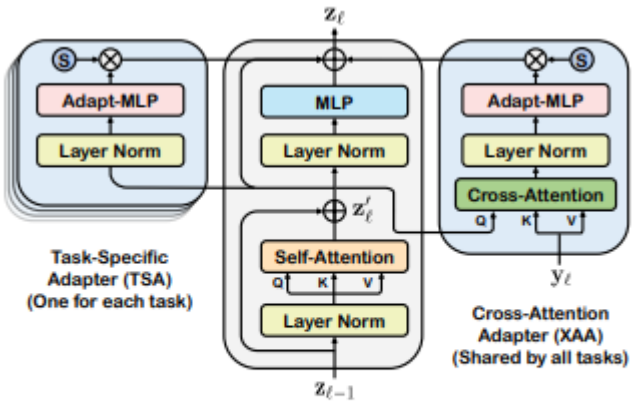


Figure 3. An illustration of a task-versatile Transformer layer equipped with two newly-introduced adapters: cross-attention adapter (XAA) and task-specific adapter (TSA).

Основой архитектуры предложенной модели выступает предтренированный CLIP [11]. Для того, чтобы приспособить модель к выполнению различных задач рекомендации, распознавания и оценки образов авторы предлагают 2 вида адаптеров: TSA – Task-Specific Adapter для обучения специфическим для каждой задачи особенностям и XAA – Cross-Attention Adapter для обеспечения возможности взаимодействиями между различными модальностями, общий для всех задач. Для TSA предлагается ввести дополнительные линейные слои (AdaptMLP) после каждого self-attention блока параллельно с основными:

$$z_l^{tsa} = s \cdot AdaptMLP(LN(z_l'))$$

где s – Обучаемый множитель.
В XAA используется дополнительный Multi-Head Cross Attention (MHXA) с группой линейных слоев после него:

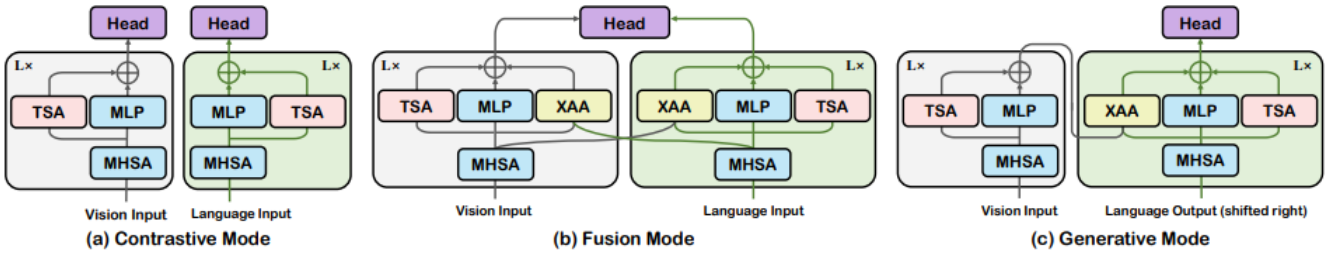
$$z_l^{xaa} = s \cdot AdaptMLP(LN(MHXA(z_l', y_l)))$$

где y_l выход self-attention слоя части сети для другой модальности.
Далее полученные z_l^{tsa} и z_l^{xaa} аггрегируются с обычным выходом следующим образом:

$$z_l = MLP(LN(z_l')) + z_l' + z_l^{tsa} + \epsilon \cdot z_l^{xaa},\ \epsilon \in \{0, 1\}$$

ϵ – барьерный множитель включающий или выключающий определенный адаптер для определенной задачи.

Рассматриваемая архитектура обучается на разные задачи в трех режимах Contrastive, Fusion и Generative с различными используемыми адаптерами и функциями потерь.



Contrastive mode:
Этот режим используется для кросс-модальных рекомендаций (XMR) — задачи выбора наиболее подходящего элемента по текстовому описанию и выбора наиболее подходящего описания элемента на изображении среди данных. Все XAA блоки отключены. Обучение производится на выборках элементов $\mathcal{X} \supseteq (\mathbf{I}, \mathbf{T}) = \{(I_1, T_1), \ldots, (I_B, T_B)\}$, сначала части сети для соответствующей модальности по отдельности применяются к элементам каждой пары, формируя 2 итоговых унимодальных представления, а потом с помощью контрастной функции потерь [7] производится максимизация схожести получившихся унимодальных представлений.

$$\mathcal{L}_{XMR} = \frac{1}{2}[\mathcal{L}_{InfoNCE}(\mathbf{T}, \mathbf{I}) + \mathcal{L}_{InfoNCE}(\mathbf{I}, \mathbf{T})]$$

$$\mathcal{L}_{InfoNCE} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(s(X_i, Y_i)/\tau)}{\sum_{j=1}^{B} \exp(s(X_i, Y_j)/\tau)}$$

где τ – обучаемая (???) температура. s – симметричная функция схожести $s(I_i, T_j) = f_\theta^{[c]}(I_i)^T \cdot f_\theta^{[c]}(T_j)$, где $f_\theta^{[c]}$ – собственно нейросеть.

Fusion mode:
Используется для субкатегориального распознавания (SCR)и направляемых текстом рекомендаций (TGIR). И XAA, и TSA блоки включены.
Задача SCR – предсказание подкатегории для данного товара, основываясь на тексте и картинке. Исходя из специфики задачи, к выходу сети дополнительно добавляется классификатор, cross-entropy-loss которого и минимизируется:

$$\mathcal{L}_{SCR} = -\mathbb{E}_{(I,T)\sim D} \log P\left(f_\theta^{[f]}(I, T)\right)$$

Для TGIR –нахождения элементов похожих по изображению на данный и соответствующих текстовому описанию – подход немного другой, поскольку необходимо получить представления отдельно для исходной картинки с текстовым описанием и целевой картинки, поэтому для $(\mathbf{I^r}, \mathbf{T})$ соответственно исходных картинок и текстовых запросов сеть запускается в $fusion$ режиме, а для целевых изображений I^t в *contrastive* режиме. Далее считается контрастная функция потерь вида

$$\mathcal{L}_{TGIR} = \mathcal{L}_{InfoNCE}((I^r, T), I^t)$$

Generative mode:
Используется, например, для генерации описания к изображениям элементов. TSA блоки включены в обеих модальностях, XAA – только image-to-text. Причем часть обрабатывающая входное изображение используется как энкодер, а вторая – как декодер в авторегрессионном режиме.

3 Теоретическая часть

Отличительной особенностью рассмотренных выше подходов является, попытка в том или ином виде моделировать взаимодействие многие ко многим в полном графе из частей внутренней структуры образов (элементов или признаков). Для моделирования взаимодействия между составными частями объекта или образа, нужно, для начала, получить некоторое их векторное представление.

В работе не будем сравнивать между собой способы векторизовать текстовые и графические представления и заранее их зафиксируем. Более того, сконцентрируемся на представлении образа как множества элементов без внутренней структуры и оставим вопрос разбиения объектов на признаки и рассмотрения их взаимодействия за рамками данной работы.

Для получения представлений текстового и графического описания каждого элемента будем использовать предобученную модель «Outfit Transformer»[12]. Под множеством объектов $\mathcal{X}$ далее будем подразумевать множество их векторных представлений $\mathcal{X} \subset \mathbb{R}^d$

Будем рассматривать задачу дополнения образа. Для ее решения, как было сказано, необходимо знать функцию совместимости. Оставим восстановление этой функции за рамками текущей работы и снова будем пользоваться ранее предложенным решением – предобученной моделью из [12], для действия этой модели на образ будем использовать обозначение введенное выше для функции оценки — $\mathcal{S}(\cdot)$. $\mathcal{S}(\cdot)$ — ограниченная непрерывно дифференцируемая функция своих аргументов, причем норма ее градиента также ограничена.

Далее нас будет интересовать исключительно случай пустого описания недостающих элементов. Учитывая постановку определенную ранее, получаем задачу следующего вида:

- **Дано:**
  $O_n \in \mathcal{O},\ |O| = n$ — исходный образ

$k \in \mathbb{N},\ k$ — количество недостающих элементов

- **Задача:**

$$\{\hat{X}_i\}_{i=1}^{k} = \underset{\{X_i\}_{i=1}^{k} \subset \mathcal{X}}{\operatorname{argmax}} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^{k}\right)$$

Для $k > 1$ и большого количества доступных элементов решение задачи полным перебором становится вычислительно невозможным (сложность алгоритма полного перебора $|\mathcal{X}|^k$), поэтому необходимо рассмотреть возможные способы приближенного решения.

3.1 Дискретный подход

Заметим, что математически задача выбора k наиболее подходящих к образу элементов может быть переформулирована как задача поиска наилучшего пути длины k в полном графе, множество вершин которого есть

$$\mathcal{X} \setminus O_n \cup \{X_{init}\},$$

где X_{init} – дополнительная начальная вершина. Для некоторого пути $X_{init}, X_1, X_2 \dots X_j$ можно рассчитать его «*оценку*» как оценку образа $O_n \cup \{X_1 \dots X_j\}$ тогда понятно, что поиск пути с максимальной оценкой эквивалентен решению поставленной выше задачи дополнения образа. Такой подход требует подсчета оценки текущего пути на каждом шаге, а значит чтобы улучшить асимптотику и получить практически вычислимый алгоритм, необходимо использовать приближенное решение задачи поиска в графе.

В качестве бейзлайна будем рассматривать два следующих жадных алгоритма:

1. «Одношаговый» $|\mathcal{X}|$ вызовов функции оценки
   - Подсчет $\mathcal{S}(O_n \cup X)$ для всех элементов $X \in \mathcal{X}$
   - Выбор элементов $X_i,\ i = \overline{1,k}$ соответствующих k максимальным значениям $\mathcal{S}(O_n \cup X)$
2. «Многошаговый» $k \cdot |\mathcal{X}|$ вызовов функции оценки
   - Подсчет $\mathcal{S}(O_n \cup X)$ для всех элементов $X \in \mathcal{X}$
   - Выбор элемента X_1 соответствующего максимальному значению $\mathcal{S}(O_n \cup X)$
   - Подсчет $\mathcal{S}(O_n \cup X_0 \cup X)$ для всех элементов $X \in \mathcal{X}$
   - Продолжение последовательных итерации до X_k

(ОФОРМИТЬ АЛГОРИТМЫ НОРМАЛЬНО!)

Возможно обобщить жадные алгоритмы и применить широко используемый, например в языковых генеративных моделях, алгоритм beam-search, однако он либо вырождается в многошаговый алгоритм выше, либо требует еще больше чем $k \cdot |\mathcal{X}|$ вызовов функции оценки для каждого образа, что ставит под вопрос практическую применимость.

3.2 Непрерывный подход

Как альтернативу приближенному перебору можно рассмотреть решение задачи во всем пространстве $\mathbb{R}^d$

$$\{\tilde{X}_i\}_{i=1}^{k} = \underset{\{X_i\}_{i=1}^{k} \subset \mathbb{R}^d}{\operatorname{argmax}} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^{k}\right)$$

Поскольку мы пользуемся предобученной моделью для подсчета функции оценки, нам доступны не только ее значения для любого образа, но и градиент в любой точке. Значит, такой постановке $\{\tilde{X}_i\}$ могут быть найдены например одной из вариаций градиентного спуска. Далее

$$\hat{X}_i = \underset{X \in \mathcal{X}}{\operatorname{argmin}} \rho(\tilde{X}_i, X),$$

где ρ – некоторая метрика, например L_p. $\mathcal{S}$ – непрерывно дифференцируемая функция с ограниченным по норме градиентом, а значит липшицева с некоторой константой M, в таком случае:

$$\sum_{i=1}^{k} \rho(\hat{X}_i, \tilde{X}_i) < \varepsilon \longrightarrow \left| \mathcal{S}\left(O_n \cup \{\tilde{X}_i\}_{i=1}^k\right) - \mathcal{S}\left(O_n \cup \{\hat{X}_i\}_{i=1}^k\right) \right| < M \cdot \varepsilon$$

Что позволяет надеяться на хорошие результаты при достаточном количестве данных.

К минусам подхода можно отнести сохраняющуюся необходимость многократного вызова функции $\mathcal{S}$ и ее градиента для решения оптимизационной задачи в $\mathbb{R}^d$, однако количество итераций градиентного спуска в этом случае по крайней мере не зависит от размеров используемого датасета.

Кроме того,

$$\sum_{i=1}^{k} \rho(\hat{X}_i, \tilde{X}_i) < \varepsilon$$

довольно сильное условие на непрерывное решение и для того, чтобы оно выполнялось, необходимо по крайней мере

$$\exists \{\hat{X}_i\}_{i=1}^k \subset \mathcal{X} : \ \mathcal{S}\left(O_n \cup \{\hat{X}_i\}_{i=1}^k\right) \geqslant \max_{\{X_i\}_{i=1}^k \subset \mathbb{R}^d} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right) - M\varepsilon$$

$$\Updownarrow$$

$$\max_{\{X_i\}_{i=1}^k \subset \mathcal{X}} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right) \geqslant \max_{\{X_i\}_{i=1}^k \subset \mathbb{R}^d} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right) - M\varepsilon$$

...ЕЩЕ ПАРУ СЛОВ О ТОМ, ЧТО МЕТОД СОМНИТЕЛЬНЫЙ...

3.3 Генерация взаимосвязанных скрытых представлений

В предложенном ниже методе полностью откажемся от вызова функции оценки на этапе применения. Для этого переформулируем задачу как поиск аппроксимации функции

$$\mathcal{F}_k : \mathcal{O} \longrightarrow \mathcal{X}^k, \quad O_n \in \mathcal{O}, \ \mathcal{F}_k(O_n) = \underset{\{X_i\}_{i=1}^k \subset \mathcal{X}}{\operatorname{argmax}} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right)$$

Композицией функций

$$F_k^\theta : \mathcal{O} \longrightarrow \mathbb{R}^d, \ F_k^\theta(O_n) = \{X_i\}_{i=1}^k$$

$$\text{и } \rho_{\mathcal{X}} : \mathbb{R}^d \longrightarrow \mathcal{X}, \ \rho_{\mathcal{X}}(X_i) = \underset{\hat{X}_i \in \mathcal{X}}{\operatorname{argmax}} \rho(X_i, \hat{X}_i)$$

где $\rho(\cdot, \cdot)$ – некоторая мера близости между аргументами. Далее, учитывая, что рассматриваемое латентное пространство $\mathbb{R}^d$ имеет большую размерность, будем рассматривать в качестве ρ косинусную близость, следуя рекомендациям из [14].

Таким образом мы свели исходную задачу к задаче генерации скрытых представлений недостающих элементов $\{\hat{X}_i\} \subset \mathbb{R}^d$, наиболее близких в смысле функции ρ к точным решениям задачи

$$\{\hat{X}_i\}_{i=1}^k = \underset{\{X_i\}_{i=1}^k \subset \mathcal{X}}{\operatorname{argmax}} \mathcal{S}\left(O_n \cup \{X_i\}_{i=1}^k\right)$$

с помощью функции F_k^θ с вектором параметров θ.

Эту функцию предлагается приближать нейронной сетью. Для обучения сети требуется набор точных решений задачи, тогда как в используемых на практике датасетах только очень малая доля всех образов имеет оценку близкую к максимально возможной, а значит необходимо предложить способ получения точного решения или его приближения за конечное время. Лучшим из рассмотренных методов, очень близким к полному перебору, является многошаговый жадный подход описанный в 3.1.

Таким образом, предлагается двухэтапный процесс построения модели генерации представлений для дополнения образа:

1. Для подмножества множества образов

$$\mathcal{O}_n = \{O^i\}_{i=1}^n \subset \mathcal{O}$$

с помощью многошагового жадного метода сгенерировать приближенные решения задачи дополнения образа

$$\mathcal{X}_n = \{\{X_j^i\}_{j=1}^k\}_{i=1}^n \subset \mathcal{X}^k$$

2. Обучить нейронную сеть для функции F_k^θ используя $\mathcal{O}_n$ и $\mathcal{X}_n$ в качестве обучающих данных, т.е.

$$\theta = \underset{\hat{\theta}}{\operatorname{argmin}} \left(\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \rho \left(X_j^i, [F_k^{\hat{\theta}}(O^i)]_j \right) \right)$$

Задача по постановке симметрична к перестановкам элементов, поэтому разумно потребовать от всех операций в архитектуре сети эквивариантности [2] к перестановкам. Кроме того, эмпирически понятно, что элементы в образе сложным образом зависимы друг от друга. Ничего больше про внутреннюю структуру данных неизвестно, поэтому в таком случае естественным будет рассмотреть применение графовых нейросетей для аппроксимации функции F_k^θ, поскольку они эквивариантны к перестановкам по построению и хорошо показывают себя в задачах требующих моделирования сложных взаимодействий между объектами схожей природы.

«MESSAGE PASSING GNN [4] ПАРУ ФОРМУЛ »
«Возможно стоит добавить GNN в обзор литературы, а здесь кратко сослаться»

.....

Используя сеть такого вида можно генерировать скрытые представления одновременно для k элементов, в отличие от всех ранее представленных подходов осуществляющих последовательное добавление в образ по одному элементу, что повышает эффективность в k раз, а также позволяет в процессе генерации моделировать взаимодействие в том числе между новыми элементами

4 Вычислительный эксперимент

4.1 Сетап (придумать нормальное слово!)

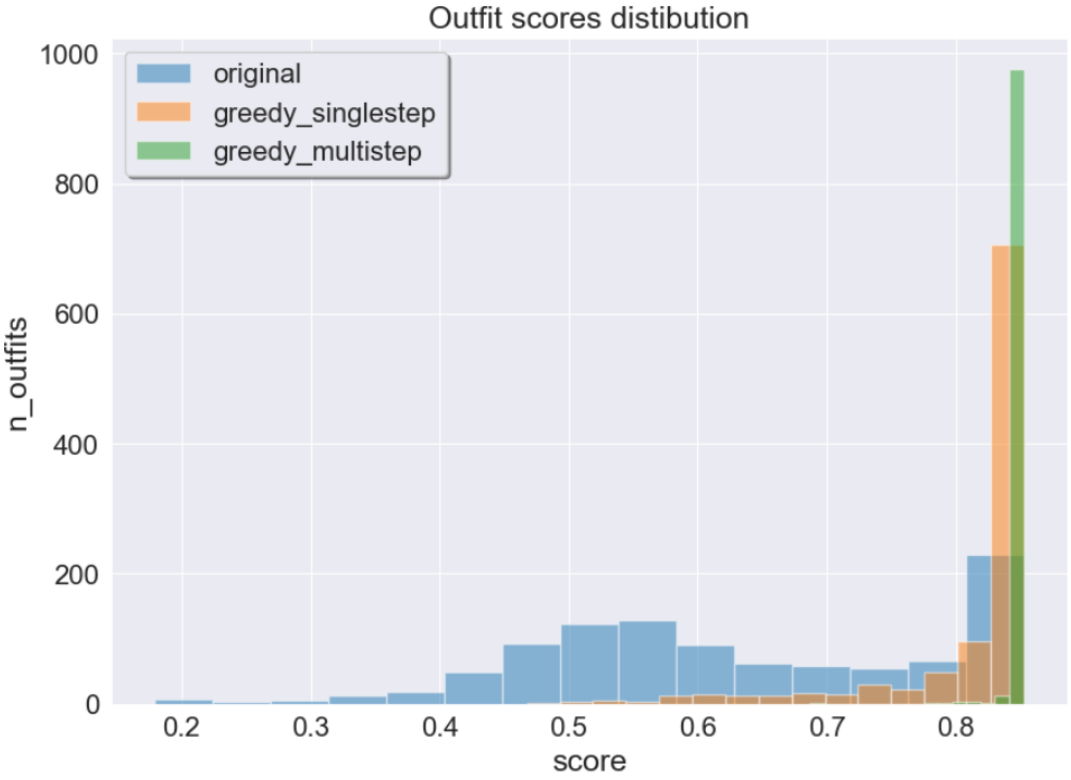
Для эксперимента будем использовать часть датасета Polyvore [6]..... Размер каждого образа $\geqslant 5$

k = 2

Архитектура: ...

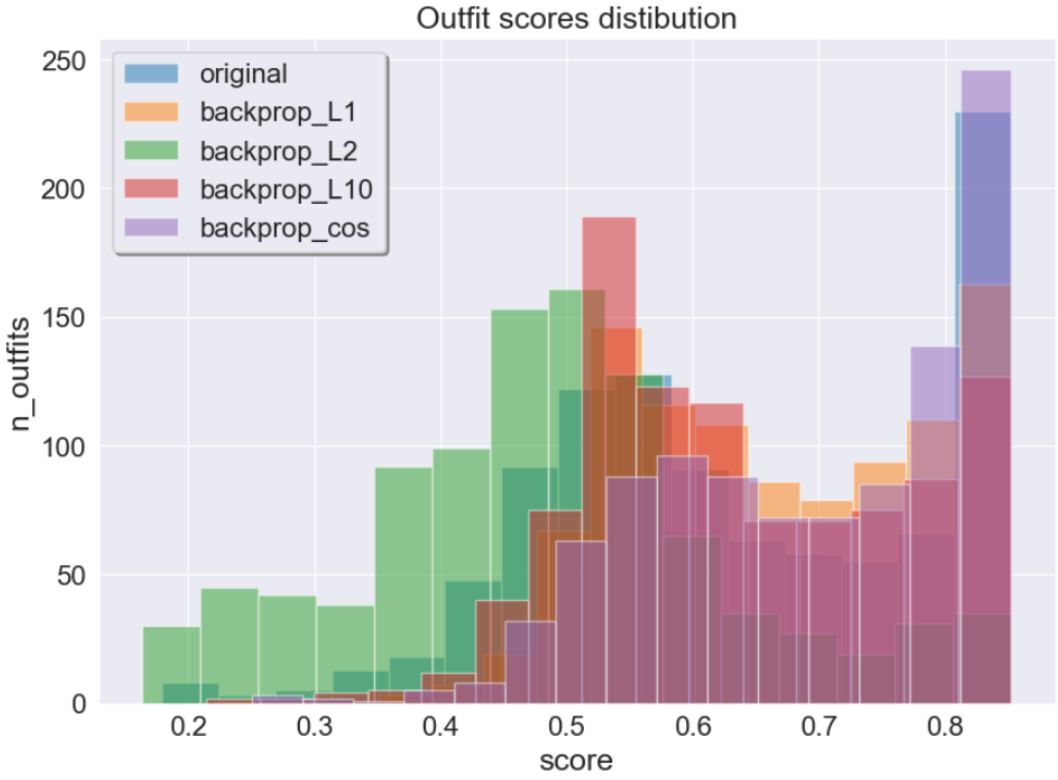
Оцениваем РАСПРЕДЕЛЕНИЕ ОЦЕНОК ОБРАЗОВ (Медиану?)

4.2 Жадные алгоримты



Катастрофически долго работает, но хорошие результаты

4.3 Непрерывный подход

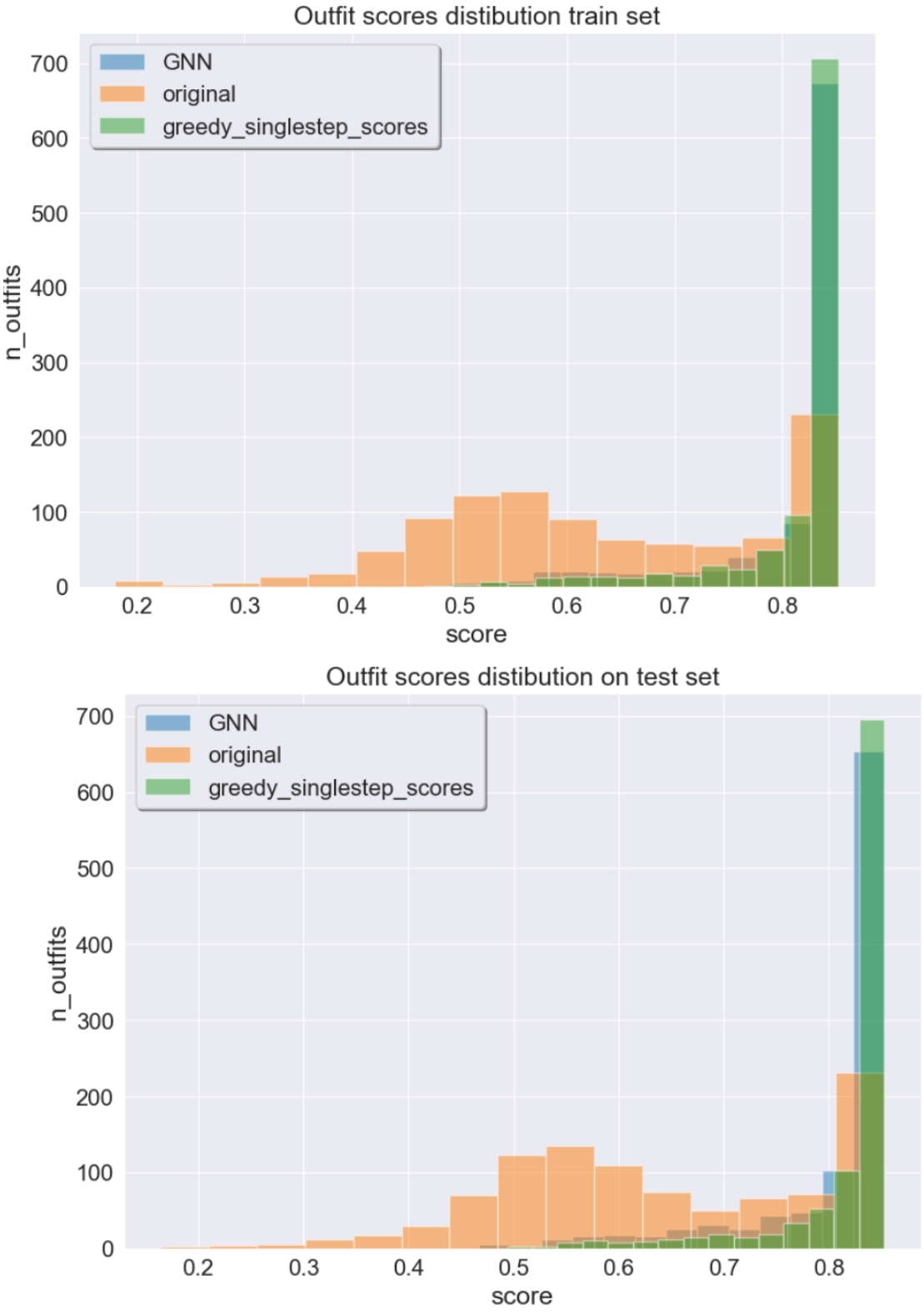


как и предполагалось работает не очень

сильно зависит от выбора нормы, который можно осуществить только перебором и эмпирическими предположениями

долго работает

4.4 Генерация скрытых представлений



Работает почти как одношаговый жадник, но быстрее на 2 порядка, что дает возможность применять на практике

Список литературы

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[2] Taco S. Cohen and Max Welling. Group equivariant convolutional networks, 2016.

[3] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.

[4] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017.

[5] Xiao Han, Xiatian Zhu, Licheng Yu, Li Zhang, Yi-Zhe Song, and Tao Xiang. Fame-vil: Multi-tasking vision-language model for heterogeneous fashion tasks, 2023.

[6] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S. Davis. Learning fashion compatibility with bidirectional lstms. *CoRR*, abs/1707.05691, 2017.

[7] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2021.

[8] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[9] Zhanghui Kuang, Yiming Gao, Guanbin Li, Ping Luo, Yimin Chen, Liang Lin, and Wayne Zhang. Fashion retrieval via graph reasoning networks on a similarity pyramid, 2019.

[10] Xingchen Li, Xiang Wang, Xiangnan He, Long Chen, Jun Xiao, and Tat-Seng Chua. Hierarchical fashion graph network for personalized outfit recommendation, 2020.

[11] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[12] Rohan Sarkar, Navaneeth Bodla, Mariya I. Vasileva, Yen-Liang Lin, Anurag Beniwal, Alan Lu, and Gerard Medioni. Outfittransformer: Learning outfit representations for fashion recommendation, 2022.

[13] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[14] Ali Seyed Shirkhorshidi, Saeed Aghabozorgi, and Teh Ying Wah. A comparison study on similarity and dissimilarity measures in clustering continuous data. *PloS one*, 10(12):e0144059, 2015.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[17] Petar Veli?kovi?, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Li?, and Yoshua Bengio. Graph attention networks, 2018.