Enhancing Semantics-Driven Recommender Systems with Visual Features

Mounir M. Bendouch[1], Flavius Frasincar[1][0000−0002−8031−758X], and Tarmo Robal[2][0000−0002−7396−8843]

[1] Erasmus University Rotterdam, Burgemeester Oudlaan 50, 3062 PA Rotterdam, The Netherlands
mbendouch@hotmail.com, frasincar@ese.eur.nl
[2] Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia
tarmo.robal@ttu.ee

Abstract. Content-based semantics-driven recommender systems are often used in the small-scale news recommendation domain, founded on the TF-IDF measure but also taking into account domain semantics through semantic lexicons or ontologies. This work explores the application of content-based semantics-driven recommender systems to large-scale recommendations on the example of movie domain. We propose methods to extract semantic features from various item descriptions, including images. In particular, we use computer vision to extract semantic features from images and use these for recommendation together with various features extracted from textual information. The semantics-driven approach is scaled up with pre-computation of the cosine similarities and gradient learning of the model. The results of the study on a large-scale MovieLens dataset of user ratings demonstrate that semantics-driven recommenders can be extended to more complex domains and outperform TF-IDF on *ROC*, *PR*, F_1, and Kappa metrics.

Keywords: Semantics-driven recommendation · Ontology · Computer vision · Visual semantic features · Large-scale recommendation.

1 Introduction

With the emergence of the Web vast amounts of information have become available with an accelerating increase [44], scaling up to 44 trillion gigabytes in 2020 [38]. This abundance of information has enabled users to explore immerse variety of content (e.g., articles, movies, music), but also introduced the problem of information overload making finding the right information difficult and time consuming. A solution for the latter problem is seen in recommender systems (RS) [28, 29], which provide mechanisms to filter and deliver content relevant to the user in the form of recommendations based on information available about the user and domain [30]. Different approaches to RS [28] exist: *collaborative filtering*, where recommendations are based on similarities between preferences of one user and preferences of others, *content-based filtering*, which recommends

items according to their content, and a combination of the two latter known as *hybrid RS* [5].

Here, we focus on content-based RS [26] operating on similarities between content items based on various extractable features, The features available depend on the item type and dataset. Although text (e.g., descriptions) is the common form of information to extract features to measure similarity, other types of information (e.g., music songs include the artist, genre, and the lyrics, movies include the actors, plot, posters) can also serve as a source of features.

A widely used technique to estimate similarity between texts is Term Frequency - Inverse Document Frequency (TF-IDF) [20], where a feature vector based on the frequency counts of terms in the text is constructed and multiplied by the inverse frequency of these terms occurrence in all text sources. The resulting vectors can then be directly compared using measures such as cosine similarity [16]. Several recommenders such as CF-IDF(+), SF-IDF(+) have taken the TF-IDF concept further to provide recommendations of news articles [4, 6, 11, 16], using concepts from domain ontologies or synsets from semantic lexicons for features instead of terms. These methods have further been extended to (Bing)-(C)SF-IDF recommenders [7, 19, 25] by including semantically related synsets or concepts, or absorbing named-entity similarities using Bing page counts.

Relying on the promising results of the latter semantics-driven RS for news articles, and encouraged by the successful scaling and porting of these methods to large scale recommendations [3], we are now eager to explore the value of semantic information extracted from items more complex than text – digital images – derived by the idea that *a picture may be worth more than a thousand words*! In this paper, we extend the extraction of semantic features from text to digital images (movie posters), and explore whether and to what extent it can contribute to recommendations. In particular, we seek to answer:

RQ1: *How to extract and apply semantic features from images for recommendation?*

RQ2: *How do semantic features from images contribute to recommendation?*

In this paper we continue and extend our previous work on semantics-driven RS [3], resulting in the following contributions:

- A method for extracting of semantic features from digital images using computer vision for the task, and the adjustment of the scaled similarity model [3] for features extracted from images.
- A proposal of novel method for large-scale semantics-driven recommendations based on concepts and synsets extracted from text, and synsets extracted from digital images.
- Demonstration that semantics-driven RS have many unexplored applications and can be utilized effectively with the proposed approach to various domains.

The rest of this paper is organized as follows. Section 2 presents related work, while Section 3 discusses data used for the research. Section 4 focuses on the recommendation methodology, and Section 5 on its evaluation. Section 6 draws conclusions.

2 Related Work

We start by reviewing the semantics-driven recommenders TF-IDF, CF-IDF, SF-IDF, and their extensions CF-IDF+, SF-IDF+ originally designed for news recommendation. These RS extract features from news article text but can be used to predict similarity between any two texts.

The TF-IDF is of interest as SF/CF-IDF(+) build on its mathematical concept. The TF-IDF [33] recommender consists of two parts, where the TF indicates how often a term occurs in a given document (higher frequencies link to higher relevancy), and the IDF captures the importance and uniqueness of a term in a collection of documents (frequent terms are considered to be common and less important). The resulting feature vector represents terms with scores, which can be compared to user vectors using similarity functions (e.g., the cosine distance). The TF-IDF score is large for terms that occur frequently in single document but not often in all other documents. A certain specified threshold value decides whether an item and the user's interest are considered similar.

The Concept Frequency - Inverse Document Frequency (CF-IDF) [16] is a variant of TF-IDF, where instead of terms concepts of domain ontology are used. The text is processed by a natural language processing (NLP) engine that performs word sense disambiguation (WSD), part-of-speech (POS) tagging, and tokenization to transform the text into a collection of concept candidates. A domain ontology containing concepts and their relationships is checked for each candidate, and if a match is found, a count is added to that concept. The use of concepts represents the domain semantics better as only relevant words of the domain are considered, and results in performance improvement over TF-IDF [16]. CF-IDF+ extends this method further by including directly related concepts in the domain ontology [11]. Each type of relationship (superclass, subclass, or instance) is given a weight to vary the overall importance of the found concepts and their related concepts. The weights are optimized by grid search.

The Synset Frequency - Inverse Document Frequency (SF-IDF) [6] is another variant of TF-IDF, which in addition to all terms looks at synonyms and ambiguous terms using a semantic lexicon (WordNet). Terms having the same meaning will be subsumed in one single concept, and therefore WSD is needed. For terms with multiple meanings, corresponding word senses are counted separately. SF-IDF+ [25] outperforms SF-IDF by including synsets that are directly related over the 27 types of semantic relationships present in WordNet, where each type has a weight optimized by a genetic algorithm.

The TF/CF/SF-IDF(+) content-based RS were originally established for news recommendation, rather small-scale recommendation domain, where they proved their efficiency for the task. The applicability of these methods to large-scale recommendation problem was proven to be successful in [3] on the example of movie domain. To enable large-scale recommendations, new methods to extract semantic features from various item descriptions were established together with a method to efficiently devise a domain ontology for the selected complex dataset in case an external ontology is not available, leveraging the need to manually construct such ontology. Further, the semantics-driven approach was

scaled up with pre-computation of the cosine similarities, reduction of dimensionality and gradient learning of the model, allowing to avoid computationally expensive operations [3]. While [3] used semantic information available in the textual form, this work extends it by including also rich semantic information available in graphical form, on the example of the movie domain and posters (digital images) available.

RS for (multi)media are of interest to many researchers due to the large diverse information available. Various approaches have been exercised to provide recommendations: a graphical model and signature-tree-based scheme over social media streams [46], knowledge graphs [18], context-aware social media recommendations [45], ontologies [1, 34], Bidirectional Encoder Representations from Transformers (BERT) [13] for conversational RS [27] with experiments on movies, books and music recommendation, Word2Vec algorithm to recommend movies [42] based on metadata (e.g., directors, actors), textual image metadata for recommending socially relevant images [21]. A comprehensive overview of RS for multimedia content is given in [12].

Convolutional neural networks (CNNs) dominate the field of computer vision in terms of performance on a variety of tasks, such as optical character recognition (OCR) [8, 9], facial recognition [23, 24], face detection [15], or to learn image shapes for recommending apparel goods [32]. On some object classification tasks [36] it can even rival human performance [31]. Guo et al. [17] used CNNs to extract features of semantic image objects, splitting image into a number of image objects, extracting the features, and then summarising the results for an image. Tuinhof et al. [37] used CNNs for image classification on fashion product images to recommend products by texture and category type features. They showed that RS purely relying on visual features are reasonable and could also be helpful in case of lacking user historical data. Yu et al. [43] on the other hand focused on goods RS based on image content represented by weighted feature model using only computationally inexpensive low-level image features such as color, texture, and shape to cut down on computation time. We use computer vision to extract visual-semantic information from movie posters.

3 Recommendation Data

As in [3], we continue to use the MovieLens 20M[3] dataset providing us 20,000,000 user ratings on a scale of 1–5 for 27,278 movies over a ten-year period from 138,493 users who had rated at least 20 movies, and acquire from the MovieLens[4] the title, year of release, genre labels, and IMDB[5] identification numbers for each movie as the item-level information for feature extraction. We use two other sources over IMDB ids: (i) OMDb[6] to query movie plots, and (ii) TMDb[7] to

[3] https://grouplens.org/datasets/movielens/20m/
[4] https://movielens.org/
[5] The Internet Movie Database, https://www.imdb.com/
[6] The Open Movie Database, https://www.omdbapi.com/
[7] The Movie Database, https://www.themoviedb.org/

Table 1. Movie information, descriptive statistics.

Data type and source	N	Missing %	Mean	Min	Max
Title (MovieLens)	27,278				
Genres (MovieLens)*	27,278		1.99	1	10
Genres (OMDb)*	27,207	0.26	2.21	1	5
Directors (OMDb)*	27,003	1.01	1.11	1	41
Plot (OMDb)**	26,327	3.49	63.49	3	1471
Writers (OMDb)*	25,831	5.30	2.41	1	35
Actors (OMDb)*	26,925	1.29	3.93	1	4
Poster (TMDb)	26,827	1.65			

* Multi-class variable, statistics reported for number of classes.
** Full text, statistics reported for number of words.

collect movie posters. We use TMDb as it provides posters freely to anyone with free user account, whereas OMDb makes them available only to patrons, and for this reason we need to use TMDb next to OMDb. TMBd provides a movie poster with sufficient resolution for 98.35% of the movies in the dataset, while OMDb provides plots for 96.51% of the movies in MovieLens. We discard movies for which no plot or poster is available. We notice that the plots are substantially shorter (in average 63 words) than typical news articles, which might reduce the amount of available semantic information. For each movie we obtain genres from MovieLens and OMDb, retaining genres from both sources, as we want to ensure no valuable information is lost due to their variability. We discard any movie that has one or more missing values in any of the variables (e.g, director, actor, poster, etc.), leaving us with the final dataset of 25,138 movies for this research. This affects only 0.83% of user ratings available. Table 1 describes the different movie-level variables we use in this research.

4 Recommendation Methodology

This section covers shortly the extraction of semantic features from the plots, described in detail in [3], followed by the extraction of semantic features form digital images. We then proceed with the recommendation method building on the existing TF/CF/SF-IDF(+) recommenders.

4.1 Feature Extraction from Textual Information

In line with TF-IDF [16], CF-IDF(+) [11, 16], and SF-IDF(+) [7] recommender systems, we extract semantic information from terms, concepts, and synsets. Variables such as genres and persons are readily available and need not to be extracted from text [3]. We use the relationships between persons (*Actors*, *Directors*, *Writers*) to construct a domain ontology, detailed in [3].

We use NLP techniques to extract terms and synsets from the plots. Using NLTK[8] package in Python 2.7, each plot is split into a set of sentences and

[8] http://www.nltk.org/

processed separately. Sentences are split into a list of words (tokens) with tokenization using known properties of words (such as they usually occur in the English dictionary). Using part-of-speech (POS), each word is tagged with the POS (e.g., noun, verb, adjective). Stop words, containing negligible semantic information, are then removed, and the Porter [39] stemming algorithm applied to each word to reduce the words to their roots and extract the terms.

Synsets are extracted using the Adapted Lesk [2] WSD algorithm on each word. WSD addresses the problem of identifying the sense of a word – the meaning in its context. Only senses that have the same POS tag as the word from the text are considered. If no sense is found, all senses with any POS are considered. The synset containing the identified sense of the word is extracted.

4.2 Feature Extraction from Images

The posters are generally made to advertise the movies and tend to show the characters and setting of the movie. For example, the poster for the movie *Toy Story* (Fig. 1) shows toys, a cowboy, and an astronaut, delivering the impression of a family movie targeted to young boys. During the study we notice that compared to the movie plot, the poster contains fewer irrelevant elements.

Each pixel in a digital image is represented by 3 colour values for red, green, and blue (RGB). Thereby, an input image of size w wide and h pixels high, can be represented as a matrix of $3 \times h \times w$ values. The most common lossless digital image compression format Portable Network Graphics (PNG) encodes pixels of an image in a 24-bit RGB palette (8 bits per colour). Computer vision libraries (e.g., OpenCV[9]) convert this to a $3 \times h \times w$ matrix of unsigned 8-bit integer values ranging from 0 to $2^8 - 1 = 255$. As most neural network libraries such as Theano[10] take floating-point numbers as inputs, the matrix is normalized by multiplying with $\frac{1}{255}$ to obtain a matrix of values in the range $[0, 1]$.

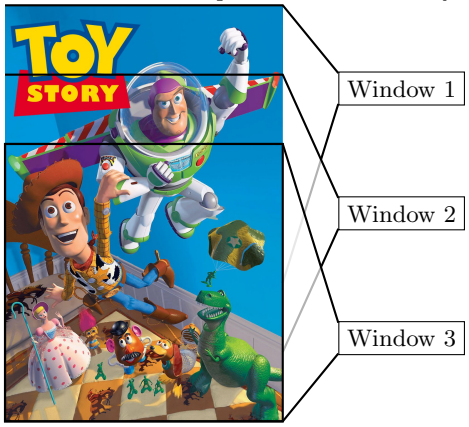
To extract semantic features from the movie posters we use techniques from computer vision – algorithms to gain high-level understanding from visual information on digital images. In particular, we use CNNs [14] that are state-of-the-art models to extract a vector of synset probabilities and a Visual-Semantic Embedding (VSE) vector from each movie poster.

In order to extract synset vectors from poster images, we exploit the VGG19 – a 19-layer deep CNN from the Visual Geometry Group of the University of Oxford [35]. VGG19 was the highest-performing submission for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[11] in 2014. ILSVRC is a competition where algorithms compete for object detection and image classification, where the challenge for the algorithms is to classify an image in $1,000$ categories that are each represented by a synset. In the tests, for 81.1% of the images the top-5 predictions included the correct class, while human performance on this metric is estimated to be around 88-95% [31]. The trained parameters

[9] http://opencv.org/
[10] http://deeplearning.net/software/theano/
[11] http://www.image-net.org/challenges/LSVRC/

Fig. 1. Crop of three windows of 224 × 224 px (right), and predicted class (synset) probabilities for each (left), feature values are the maximum probabilities of each synset.

Comic book	0.8770
Book jacket	0.0336
Toyshop	0.0294
Shoe shop	0.0147
Jigsaw puzzle	0.0098
Bookshop	0.0042
Puck	0.0031
Tray	0.0030
Confectionery	0.0016
Tobacco shop	0.0013



for this model are publicly available[12]. VGG19's convolutional layers each have a filter size of 3×3 and the input to each of those layers is zero-padded with $p = 1$ such that the outputs are of equal spatial dimensions. Down-sampling occurs only through max-pooling layers. Two fully connected layers are added and connected to a 1,000 dimensional softmax output layer. As substantial semantic content of the posters can be described by the objects that can be recognized from them, we can use VGG19 to extract meaningful synset vectors. The model takes a 224×224 colour image as input, represented as a $224 \times 224 \times 3$ matrix of RGB pixel values, therefore poster images are down-scaled to the width of 224 px keeping the aspect ratio. The height is then still larger than 224 but never larger than 3×224, so we can take 3 vertically overlapping 224×224 windows of the poster as inputs to ensure every part of the image is covered. Fig. 1 exemplifies these windows on the poster for the movie *Toy Story* with identified synsets and their probabilities. VGG19 outputs a vector of $1,000$ probabilities, one for each synset. We evaluate the model on each window, after which we take the maximum of the 3 output values for each class (synset). We apply this procedure to the posters to obtain feature vectors of $1,000$ synset values.

The synset values returned by VGG19 are intended to classify an image and do not necessarily describe a poster fully. We therefore also consider another approach called Visual-Semantic Embedding (VSE) [22] that has been used for the challenge of image captioning [40], where the aim is to generate a natural language caption that best describes the content of an input image (i.e., translating images to text). This is done by mapping the image and the sequence of words of a caption to a common feature space – visual-semantic space – in which semantic distances between an image and a caption can be calculated. From this distance metric the semantic similarity between an image and a caption can be

[12] http://www.robots.ox.ac.uk/~vgg/research/

estimated and the nearest-neighbour caption can be returned. Our goal to represent the posters in a semantic space can be considered equivalent to mapping them to a visual-semantic embedding.

The embeddings can be learned with knowledge of pairs of images and their captions. In visual-semantic space, an image and its caption should be close. Let us define this closeness as the cosine similarity between the image's embedding $\vec{m} \in \mathbb{R}^n$ and the embedding of the caption $\vec{c} \in \mathbb{R}^n$. In a properly constructed visual-semantic space, for the image and its caption, $cos(\vec{m}, \vec{c})$ should be relatively high. Reversely, a non-descriptive caption c_r should lead to a relatively low $cos(\vec{m}, \vec{c_r})$. As the image and the caption are mapped to the same visual-semantic space, we can also expect that the more semantically similar $poster_1$ and $poster_2$ are, the higher their $cos(\vec{m_1}, \vec{m_2})$ – which is exactly the aim of our semantics-driven recommender.

Mapping an image to a visual-semantic space is done in [22] by a form of transfer learning [41], where the 4,096 visual features from the second-to-last layer of the pre-trained VGG19 model are transferred to a new model in which they are multiplied by a matrix of trainable weights θ_m, resulting in an embedding vector $\vec{m} \in \mathbb{R}^n$. Transfer learning simplifies the problem from learning the visual-semantic embedding from raw pixels to learning it from high-level visual features trained on the ImageNet Challenge.

Another trainable neural network with weights θ_c transforms the text of the caption in an embedding vector $\vec{c} \in \mathbb{R}^n$. We denote a non-matching caption for image embedding $\vec{m}$ as $\vec{c_r}$ and a non-matching image for caption embedding $\vec{c}$ as $\vec{m_r}$. All weights $\theta = \{\theta_m, \theta_c\}$ are trained simultaneously to minimize the following pairwise ranking loss:

$$\begin{aligned} &\sum_m \sum_r max\{0, \alpha - s(\vec{m}, \vec{c}) + s(\vec{m}, \vec{c_r})\} \\ +&\sum_c \sum_r max\{0, \alpha - s(\vec{c}, \vec{m}) + s(\vec{c}, \vec{m_r})\} \end{aligned} \tag{1}$$

where $s(\vec{m}, \vec{c}) = \vec{m} \cdot \vec{c}$ is the scoring function. As [22], we first scale the embedding vectors $\vec{m}$ and $\vec{c}$ to unit norm, making s equivalent to cosine similarity $s(\vec{m}, \vec{c}) = cos(\vec{m}, \vec{c})$. For the purpose of extracting semantic features from the movie posters, we are interested in the VSE $\vec{m}$ of the images. The authors of [22] have made an embedding matrix to generate 1,024-dimensional visual-semantic embeddings publicly available[13]. This matrix was trained to optimize Eq. 1 on public image captioning datasets. Our procedure consists of using this pre-trained embedding matrix on the 4,096-dimensional VGG19 visual feature vectors of the movie posters to obtain their visual-semantic embeddings.

The VSE vectors have a more solid theoretical foundation compared to the synset vectors, being derived from a state-of-the-art method whose purpose is to translate images to text. This is a more direct way of achieving our goal of extracting semantic features, and we expect this to improve recommender performance compared to VGG19 synset vectors. The VSE method however has

[13] https://github.com/ryankiros/visual-semantic-embedding

a disadvantage – the features are hidden and have no natural interpretation, making it complicated to link them to an ontology or semantic lexicon.

4.3 Scaling Visual Features

The 1,000 synset values (VGG19) and the 1,024 VSE values extracted from the posters could benefit from scaling as we expect that some features are more relevant to the content of the movies and thus should play a larger role in the cosine distance, therefore scaled higher. We have little information about the relevance of each of the 1,000 synsets, and even less about the 1,024 visual-semantic features. We learn 1,000 scales for the synsets and 1,024 scales for the visual-semantic features simultaneously with optimizing the model through stochastic gradient descent (SGD). We apply the established similarity model scaling [3] also to synsets and visual-semantic features extracted from posters. Denoting the scale as $\vec{c_i}$, if it applies to the i-th feature type t_i, leads $\vec{c_i} \in \mathbb{R}^{1,000} \Leftrightarrow t_i = VGG19$ and $\vec{c_i} \in \mathbb{R}^{1,024} \Leftrightarrow t_i = VSE$. The user-profile vector u_i and the unseen item vector $\vec{v_i}$ are then scaled through $\vec{c_i} \circ \vec{u_i}$ and $\vec{c_i} \circ \vec{v_i}$ respectively, with $\circ$ the element-wise product. These resulting scaled vectors are used in the cosine. We restrict $\vec{c_i} \geqslant 0$ and $\sum \vec{c_i} = 1$ to avoid the over-parametrization caused by $cos(\lambda\vec{u}, \lambda\vec{v}) = cos(\vec{u}, \vec{v})\ \forall\lambda \neq 0$. Further, we use both the scaled vectors and unscaled original vectors in the model for comparison. Table 2 lists all used feature types.

Table 2. Characterization of used feature types

i	Feature type t_i	Extracted from	Dataset	n_i*	m_i**
1	Directors	Variable	OMDb	12,231	4
2	Actors	Variable	OMDb	45,393	4
3	Writers	Variable	OMDb	27,415	4
4	MovieLens genres	Variable	MovieLens	19	1
5	OMDb genres	Variable	OMDb	27	1
6	Terms	Plot	OMDb	48,083	1
7	Synsets	Plot	OMDb	69,977	19
8	VGG19	Poster	TMDb	1,000	1
9	VSE	Poster	TMDb	1,024	1

* #Features i.e., length of feature vectors. ** #Relations.

To learn scaling for visual feature types, we use the similarity model (Eq. 2) established in [3], where s_i is part similarity (here cosine similarity) and w_i its weight, $\vec{u_i}$ user-profile feature vector, $\vec{v_i}$ unseen item feature vector, $\vec{q_i}$ vector of relation weights, $\vec{U_i}$ user feature matrix, and $\vec{V_i}$ feature matrix for unseen item:

$$sim = \sum_{i=1}^{k} w_i s_i = \sum_{i=1}^{k} w_i \cdot cos(\vec{u_i}, \vec{v_i}) = \sum_{i=1}^{k} w_i \frac{\vec{q_i}(U_i V_i^\top)\vec{q_i}^\top}{\sqrt{\vec{q_i}(U_i U_i^\top)\vec{q_i}^\top}\sqrt{\vec{q_i}(V_i V_i^\top)\vec{q_i}^\top}} \quad (2)$$

In the similarity model [3] we insert $\vec{u_i} \leftarrow (\vec{c_i} \circ \vec{u_i})$ and $\vec{v_i} \leftarrow (\vec{c_i} \circ \vec{v_i})$, where $\vec{c_i} \in \mathbb{R}^{n_i}$ is the learnable scaling, $\vec{u_i} = U_i$, and $\vec{v_i} = V_i$, because the number of relations $m_i = 1$ for these feature types. We restrict $\sum_{l=1}^{m_i} \vec{q}_{il} = 1$, making $\vec{q_i} = 1$ redundant, and rewrite part-similarity model s_i as given by Eq. 3:

$$sim = \sum_{i=1}^{k} w_i s_i = \sum_{i=1}^{k} w_i \frac{(\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{v_i})^\top}{\sqrt{(\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{u_i})^\top}\sqrt{(\vec{c_i} \circ \vec{v_i})(\vec{c_i} \circ \vec{v_i})^\top}} \tag{3}$$

The scaling c_i has n_i optimizable parameters and therefore by definition the model is at least n_i-dimensional – this is irreducible. However, when we want to re-use the learned scaling, we can pre-compute $\vec{c_i} \circ \vec{u_i}$ and $\vec{c_i} \circ \vec{v_i}$ because the scaling is known and fixed in that case. Then we can redefine $\vec{u_i} = \vec{c_i} \circ \vec{u_i}$ and $\vec{v_i} = \vec{c_i} \circ \vec{v_i}$ and use our efficient model [3] with pre-computed $U_i U_i^\top$, $U_i V_i^\top$, and $V_i V_i^\top$.

5 Experiments and Results

The similarity model is directly trained on pairs of user-profiles and corresponding unseen items to recommend items for which the predicted similarity is above a certain threshold value, following the procedure established in [3]. The stochastic gradient descent (SGD) is applied on the gradient of the similarity model.

An item is considered to be liked by a user if it is rated with a score ≥ 4.5, otherwise disliked, resulting in an average proportion of 19.12% liked items and 20.9 liked items per user. Further, we shuffle the order of users in our dataset and take the first 1,000 as the test set for evaluation, the following 1,000 as the validation set for the similarity model (including early stopping while training), and the rest 136,493 as the training set to optimize the similarity model.

An observation is a pair of user-profile and unseen item. User-profiles are constructed by sampling $p = 5$ liked items from a user. For each observation the feature matrices $U_i V_i^\top$, $U_i U_i^\top$, and $V_i V_i^\top$ are constructed from the X_i pre-computed data. The $V_i V_i^\top$ are retrieved as blocks of X_i, while $U_i V_i^\top$ and $U_i U_i^\top$ are constructed from sums of p blocks.

For the train and validation sets, the unseen items are defined as all items not in the user-profile. For each user-profile, we sample a liked or a disliked item with equal probability such that we obtain balanced train and validation sets with $E(y) = 0.5$. Each observation is therefore a random user-profile and item, sampled from a random user. We sample 100 batches of 1,024 validation observations and 1,374 training batches of 1,024 observations, for totals of 102,400 and 1,406,976 respectively.

To allow the test set to reflect a realistic recommendation setting, we sample the $p = 5$ user-profile items by shuffling all rated items and then iteratively discarding the first item, adding it to the user-profile if it is liked. We stop as soon as we have obtained $p = 5$ liked items. All discarded liked and disliked items are then considered to be seen. Thus, we simulate the situation when a RS detects that a user has liked $p = 5$ items. We require the unseen items to

contain at least one liked and one disliked item to able to measure performance, leaving us with 809 eligible user-profiles from the 1,000 test users. We then construct observations for the user-profile with each unseen item, and save these in a separate batch for each user. The test data is therefore composed of 809 batches of varying sizes, namely the number of unseen items. The comparison between the predicted scores and the actual likes forms the basis of performance measurement. The similarity model is trained with SGD and follows the method (Algorithm 1) described in [3].

We demonstrate the value of semantics-driven recommendations by comparison to the traditional TF-IDF recommender (denoted as T) as a baseline with terms from plots. Our version of SF-IDF+ based on synsets from plots is called S, modified CF-IDF+ holding 5 concept feature types (Directors, Actors, Writers, and genres from MovieLens and OMDb) and operating on the ontology as C, VGG19 as VG, and VSE as VS. When the visual feature scaling of VG or VS is learned (optimized) together with the rest of the parameters, the component is denoted VG_L or VS_L respectively. When the VG scaling is pre-trained in another model and transferred to this model, we denote the component VG_R (each of the 10 restarts uses a pre-trained scaling from a different restart of VG_L) or VG_A (each of the 10 restarts uses the same pre-trained scaling – the average scaling over all 10 restarts of VG_L). Our proposed semantics-driven model is called C+S+VG_A, combining the concepts (C) with synsets from plots (S) and posters (VG), where the scaling for the VGG19 synsets is transferred from the average of the 10 optimized VG_L models. Table 3 lists all models used. We test the proposed C+S+VG_A model against the TF-IDF benchmark and against all alternative models.

Table 3. Models and their optimization results, averages over 10 random restarts; n=102,400 validation and n=1,406,976 train observations. Scaling transferred from VG_L for C+S+VG_R and C+S+VG_A.

Model	k*	θ**	Logloss***		Training time****		
			Valid.	Train	Epochs	Secs/Epoch	Minutes
T benchmark)	1	2	0.6896	0.6900	10.0	6.4	1.1
C	5	18	0.6815	0.6826	11.9	10.3	2.0
S	1	21	0.6912	0.6914	11.0	14.7	2.7
C+S	6	38	0.6812	0.6822	11.0	22.7	4.2
VG	1	2	0.6924	0.6925	9.4	6.4	1.0
VS	1	2	0.6930	0.6931	8.1	6.3	0.9
VG_L	1	1,002	0.6797	0.6797	26.4	87.3	38.4
VS_L	1	1,026	0.6779	0.6777	39.3	64.4	42.2
C+S+VG	7	39	0.6810	0.6820	11.7	23.3	4.5
C+S+VG_L	7	1,039	0.6681	0.6694	35.7	117.0	69.7
C+S+VG_R	7	39	0.6708	0.6716	9.4	23.8	3.8
C+S+VG_A	7	39	0.6671	0.6680	10.4	23.0	4.0

* Number of feature types (part-similarities) ** Number of parameters.
*** Minimum over all epochs. **** Until early stopping.

We start by describing the results for the computational load of the optimization procedure implemented in Python 2.7 using Keras[14] and Theano[15] libraries, with calculations performed on a regular desktop PC with NVIDIA GTX1060 CPU enabling efficient parallel computations of the gradient updates in batches of 1,024 observations. To optimize C+S+VG_A and C+S+VG_R, we first optimize the VG_L model, extract the visual scaling from the 10 restarts, and pre-compute the VGG19 dot-products with this scaling. Table 3 presents the optimization results. We find training within reasonable limits, taking fewer than 70 minutes for even the heaviest model C+S+VG_L. The impact of our scalability method is reflected in a 15x reduction in seconds per epoch of the VG model, which uses pre-computed dot-products, compared to its VG_L counter-part using the traditional approach. Although the VS_L model with visual-semantic embeddings has 1,024 features compared to 1,000 synset features for the VG_L model, it takes about 1.5x as many epochs to converge and results in a slightly better logloss. The sparsity of the VGG19 vectors compared to the VSE vectors could have been a factor in this. For the unscaled visual vectors we see the opposite, as VG needs slightly more epochs and results in a lower loss.

We continue with the comparison between the predicted scores and the actual likes, which forms the basis of performance measurement expressed through area under curve (AUC) for the precision-recall (PR) and receiver operating characteristic (ROC) curve, F_1-measure, and Cohen's kappa [10] coefficient κ. Even though we do not directly optimize for these metrics, a lower logloss results in higher test performance (Table 4). Table 4 presents the analysis of performance metrics over all models, showing that concepts alone (C) are more informative than both synsets (S) and terms (T), while the combination of C+S [3] outperforms T on all metrics. The inclusion of features captured from poster images further improves (depending on method) the recommendation, as the proposed C+S+VG_A model outperforms C+S, and thereby also the benchmark T.

Comparing the visual feature models we see the unscaled VG outperforms VS, indicating the 1,000 synset feature values we extracted from the posters are more suitable for recommendation than the 1,024-dimensional visual-semantic embeddings. Optimized scaling results in a large performance increase: from an AUC(ROC) of 0.508 to 0.605 for VS_L and from 0.525 to 0.605 for VG_L. Under learned scaling VS_L rivals VG_L on some metrics, and closes the gap on AUC(ROC). These results indicate that the visual-semantic embeddings do not improve recommender performance over the synset vectors.

When the mean optimized scales of VG_L are transferred to the C+S+VG_R model, it strongly outperforms its unscaled version C+S+VG and all other recommenders without learned scaling. When we collect the average VG scale over 10 random restarts of VG_L and transfer this to C+S+VG_A, we see that it strongly outperforms all other models.

The proposed C+S+VG_A recommender model outperforms the traditional benchmark TF-IDF by a large margin on all metrics. Average AUC(ROC) im-

[14] (https://keras.io)
[15] https://pypi.org/project/Theano/

Table 4. Performance on test set, $n = 809$ users, averages over 10 random restarts.

Models	*AUC*		F_1		κ	
	ROC	PR	$\min_r$	$\max_r$	$\min_r$	$\max_r$
T (TF-IDF, benchmark)	0.535	0.324	0.413	0.479	0.041	0.200
C	0.567	0.358	0.419	0.507	0.081	0.249
S (SF-IDF+)	0.531	0.319	0.411	0.477	0.038	0.198
C+S	0.570	0.361	0.419	0.509	0.083	0.251
VG	0.525	0.308	0.415	0.476	0.036	0.189
VS	0.508	0.299	0.415	0.472	0.018	0.176
VG_L	0.605	0.347	0.429	0.519	0.110	0.262
VS_L	0.605	0.370	0.422	0.517	0.115	0.268
C+S+VG	0.574	0.362	0.419	0.510	0.087	0.253
C+S+VG_L	0.624	0.385	0.431	0.531	0.131	0.289
C+S+VG_R	0.624	0.386	0.432	0.532	0.128	0.286
C+S+VG_A	0.634	0.391	0.435	0.537	0.137	0.298

proves from 0.531 to 0.634, and AUC(PR) from 0.324 to 0.391. We improve $\min_r(F_1)$ from 0.413 to 0.435, and $\max_r(F_1)$ from 0.479 to 0.537. Kappa metrics are improved from 0.038 to 0.137 and from 0.198 to 0.298 for $\min_r(\kappa)$ and $\max_r(\kappa)$ respectively. Given the separately pre-trained visual scaling, we can optimize the model with the scalable approach using pre-computed dot-products just in 4-5 minutes. It is neither necessary to train the scaling together with the model as a whole, nor to directly optimize on the final performance metrics.

6 Conclusion

In this paper we continued our work on scaling content-based semantics-driven RS to large-scale recommendation task, and extended the approach to include features delivered by computer vision. The paper delivers the second phase of our work earlier work [3]. While previously [3] we showed that semantic information can be extracted not only from articles but also from information of different nature represented as text, established a method for virtual ontology construction, when suitable domain ontology is not readily available, and showed that effective scales can be found through direct optimization of the logloss within minutes on consumer-grade hardware, we now demonstrated that rich semantic information can be extracted from digital images to further improve recommendations. Through a reformulation of how related features are combined, we were able to pre-compute the computationally expensive operations of the cosine similarities and reduced the dimensionality of the similarity model by several orders of magnitude. Overall, we showed that semantics-driven RS can be extended to more complex domains with high-quality recommendations on an extremely large scale.

The proposed semantics-driven recommender C+S+VG_A enhanced with visual features strongly outperformed the baseline TF-IDF, and all other models on *ROC*, *PR*, F_1, and κ, even though it was not directly optimized on these

metrics but on a cross-entropy loss function that allowed for efficient gradient-based optimization. We showed that semantics-driven RS can be extended to more complex domains with high-quality recommendations on an extremely large scale. The visual synsets extracted from images do not have to be disambiguated but can perhaps be augmented with related synsets from WordNet. The convincing success of learned feature scaling introduces the possibility of models with greater degrees of freedom, especially since the short training time on commodity hardware means that still larger datasets can be utilized.

References

1. Arafeh, M., Ceravolo, P., Mourad, A., Damiani, E., Bellini, E.: Ontology based recommender system using social network data. Future Generation Computer Systems **115**, 769–779 (2021)
2. Banerjee, S., Pedersen, T.: An adapted lesk algorithm for word sense disambiguation using WordNet. In: Gelbukh, A. (ed.) Computational Linguistics and Intelligent Text Processing. pp. 136–145. Springer, Berlin, Heidelberg (2002)
3. Bendouch, M.M., Frasincar, F., Robal, T.: Addressing scalability issues in semantics-driven recommender systems. In: IEEE/WIC/ACM International Conference on Web Intelligence (WI-IAT '21). Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3486622.3493963
4. Brocken, E., Hartveld, A., de Koning, E., van Noort, T., Hogenboom, F., Frasincar, F., Robal, T.: Bing-CF-IDF+: A semantics-driven news recommender system. In: Giorgini, P., Weber, B. (eds.) Advanced Information Systems Engineering. pp. 32–47. Springer, Cham (2019)
5. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction **12**(4), 331–370 (2002)
6. Capelle, M., Frasincar, F., Moerland, M., Hogenboom, F.: Semantics-based News Recommendation. In: Proc. of the 2nd International Conference on Web Intelligence, Mining and Semantics. WIMS '12, ACM, New York, NY, USA (2012)
7. Capelle, M., Moerland, M., Hogenboom, F., Frasincar, F., Vandic, D.: Bing-SF-IDF+: A Hybrid Semantics-Driven News Recommender. In: Proc. of the 2015 ACM Symposium on Applied Computing. p. 732–739. SAC '15, ACM, New York, NY, USA (2015)
8. Cireşan, D.C., Meier, U., Masci, J., Gambardella, L.M., Schmidhuber, J.: Flexible, high performance convolutional neural networks for image classification. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Two. pp. 1237–1242. IJCAI'11, AAAI Press (2011)
9. Ciregan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3642–3649 (June 2012)
10. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement **20**(1), 37–46 (1960)
11. de Koning, E., Hogenboom, F., Frasincar, F.: News Recommendation with CF-IDF+. In: 30th Intl Conference on Advanced Information Systems Engineering (CAiSE 2018). LNCS, vol. 10816, pp. 170–184. Springer, Cham (2018)
12. Deldjoo, Y., Schedl, M., Cremonesi, P., Pasi, G.: Recommender systems leveraging multimedia content. ACM Comput. Surv. **53**(5) (2020)

13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, MN (2019)
14. Egmont-Petersen, M., de Ridder, D., Handels, H.: Image Processing with Neural Networks—a review. Pattern Recognition **35**(10), 2279–2301 (2002)
15. Farfade, S.S., Saberian, M.J., Li, L.J.: Multi-view face detection using deep convolutional neural networks. In: 5th ACM on International Conference on Multimedia Retrieval. p. 643–650. ICMR '15, ACM, New York, NY, USA (2015)
16. Goossen, F., IJntema, W., Frasincar, F., Hogenboom, F., Kaymak, U.: News Personalization Using the CF-IDF Semantic Recommender. In: Proceedings of the 1st International Conference on Web Intelligence, Mining and Semantics. WIMS '11, ACM, New York, NY, USA (2011)
17. Guo, G., Meng, Y., Zhang, Y., Han, C., Li, Y.: Visual semantic image recommendation. IEEE Access **7**, 33424–33433 (2019)
18. Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., He, Q.: A survey on knowledge graph-based recommender systems. IEEE Transactions on Knowledge and Data Engineering pp. 1–1 (2020)
19. van Huijsduijnen, L.H., Hoogmoed, T., Keulers, G., Langendoen, E., Langendoen, S., Vos, T., Hogenboom, F., Frasincar, F., Robal, T.: Bing-CSF-IDF+: A semantics-driven recommender system for news. In: Darmont, J., Novikov, B., Wrembel, R. (eds.) New Trends in Databases and Information Systems. pp. 143–153. Springer, Cham (2020)
20. Jones, K.S.: A Statistical Interpretation of Term Specificity and its Application in Retrieval. Journal of Documentation **28**(1), 11–21 (1972)
21. Karlsen, R., Elahi, N., Andersen, A.: Personalized recommendation of socially relevant images. In: Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics. WIMS '18, ACM, New York, NY, USA (2018)
22. Kiros, R., Salakhutdinov, R., Zemel, R.S.: Unifying Visual-Semantic Embeddings with Multimodal Neural Language models. CoRR **abs/1411.2539** (2014), `http://arxiv.org/abs/1411.2539`
23. Lawrence, S., Giles, C.L., Tsoi, A.C., Back, A.D.: Face Recognition: a Convolutional Neural-Network Approach. IEEE Transactions on Neural Networks **8**(1), 98–113 (1997)
24. Matsugu, M., Mori, K., Mitari, Y., Kaneda, Y.: Subject Independent Facial Expression Recognition with Robust Face Detection using a Convolutional Neural Network. Neural Networks **16**(5–6), 555 – 559 (2003), advances in Neural Networks Research: IJCNN '03
25. Moerland, M., Hogenboom, F., Capelle, M., Frasincar, F.: Semantics-based News Recommendation with SF-IDF+. In: Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics. WIMS '13, ACM, New York, NY, USA (2013)
26. Pazzani, M.J., Billsus, D.: Content-Based Recommendation Systems, pp. 325–341. Springer Berlin Heidelberg, Berlin, Heidelberg (2007)
27. Penha, G., Hauff, C.: What does BERT know about books, movies and music? probing bert for conversational recommendation. In: 14th ACM Conference on Recommender Systems. p. 388–397. RecSys '20, ACM, New York, NY, USA (2020)
28. Rafsanjani, A.H.N., Salim, N., Aghdam, A.R., Fard, K.B.: Recommendation Systems: A Review. International Journal of Computational Engineering Research **3**(5), 47–52 (2013)

29. Ricci, F., Rokach, L., Shapira, B.: Recommender Systems Handbook. Springer, Boston, MA, USA (2015)
30. Robal, T., Haav, H., Kalja, A.: Making Web users' domain models explicit by applying ontologies. In: Advances in Conceptual Modeling - Foundations and Applications, ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM, SeCoGIS. LNCS, vol. 4802, pp. 170–179. Springer, Berlin (2007)
31. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.S., Berg, A.C., Li, F.: Imagenet Large Scale Visual Recognition Challenge. CoRR **abs/1409.0575** (2014), `http://arxiv.org/abs/1409.0575`
32. Saga, R., Duan, Y.: Apparel goods recommender system based on image shape features extracted by a CNN. In: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 2365–2369 (2018)
33. Salton, G., Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. Information Processing and Management **24**(5), 513–523 (1988)
34. Sheridan, P., Onsjö, M., Becerra, C., Jimenez, S., Dueñas, G.: An ontology-based recommender system with an application to the Star Trek television franchise. Future Internet **11**(9), 182 (2019)
35. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-scale Image Recognition. CoRR **abs/1409.1556** (2014), `http://arxiv.org/abs/1409.1556`
36. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going Deeper with Convolutions. CoRR **abs/1409.4842** (2014), `http://arxiv.org/abs/1409.4842`
37. Tuinhof, H., Pirker, C., Haltmeier, M.: Image-based fashion product recommendation with deep learning. In: Nicosia, G., Pardalos, P., Giuffrida, G., Umeton, R., Sciacca, V. (eds.) Machine Learning, Optimization, and Data Science. pp. 472–481. Springer International Publishing, Cham (2019)
38. Turner, V., Gantz, J.F., Reinsel, D., Minton, S.: The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. International Data Corporation, White Paper, IDC_1672 (2014)
39. Van Rijsbergen, C., Robertson, S., Porter, M.: New Models in Probabilistic Information Retrieval. British Library research & development report, Computer Laboratory, University of Cambridge, Cambridge, England (1980)
40. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and Tell: A Neural Image Caption Generator. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
41. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A Survey of Transfer Learning. Journal of Big Data **3**(1), 9 (2016)
42. Yoon, Y.C., Lee, J.W.: Movie recommendation using metadata based word2vec algorithm. In: 2018 International Conference on Platform Technology and Service (PlatCon). pp. 1–6. IEEE (2018)
43. Yu, L., Han, F., Huang, S., Luo, Y.: A content-based goods image recommendation system. Multimedia Tools and Applications **77**, 4155–4169 (2018)
44. Zhang, G.Q., Zhang, G.Q., Yang, Q.F., Cheng, S.Q., Zhou, T.: Evolution of the Internet and its Cores. New Journal of Physics **10**(12), 123027 (2008)
45. Zhou, X., Qin, D., Chen, L., Zhang, Y.: Real-time context-aware social media recommendation. The VLDB Journal **28**(2), 197–219 (2019)
46. Zhou, X., Qin, D., Lu, X., Chen, L., Zhang, Y.: Online social media recommendation over streams. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). pp. 938–949. IEEE, Piscataway, New Jersey (2019)