A Visual-Semantic Approach for Building Content-Based Recommender Systems

Mounir M. Bendouch[a], Flavius Frasincar[a], Tarmo Robal[b,*]

[a] *Erasmus University Rotterdam, PO Box 1738, 3000 DR, Rotterdam, the Netherlands*
[b] *Tallinn University of Technology, Akadeemia tee 15A, 12618, Tallinn, Estonia*

Abstract

The topic of recommending items based on multimodal content has been addressed to a limited extent, and yet this could be a potential solution to the data bottleneck problem. Content-based semantics-driven recommender systems are often applied 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. In this work, we explore the application of content-based semantics-driven recommender systems to large-scale recommendations and focus on using both textual information and visual information to recommend items that have multimodal content. We propose methods to extract semantic features from various item descriptions, including digital images. In particular, we use computer vision to extract visual-semantic features from images and use these for movie recommendation together with various features extracted from textual information. The visual-semantic 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 visual-semantic recommenders suitable for more complex domains than news recommendation, and which outperform TF-IDF-based recommenders on *ROC*, *PR*, F_1, and Kappa metrics.

Keywords: semantics-driven recommendation, ontology, computer vision, visual semantic features, large-scale recommendation

*Corresponding author
Email addresses: mbendouch@hotmail.com (Mounir M. Bendouch), frasincar@ese.eur.nl (Flavius Frasincar), tarmo.robal@ttu.ee (Tarmo Robal)

1. Introduction

With the emergence of the Web, immense amounts of information have become available with an accelerating speed in increase [59], scaling up to 44 trillion gigabytes in 2020 [50]. This abundance of information has on the one hand enabled users to explore an enormous variety of content (e.g., articles, movies, and music), and be the producers of such information. By this, virtually every niche and taste for content has become just mouse-clicks away. On the other hand, this abundance of choice has introduced the problem of information overload, which has made the process of finding the right information difficult and time-consuming.

A solution for the latter problem is seen in recommender systems (RS) [38, 39], which go beyond plain information retrieval systems, such as search engines, and provide mechanisms to filter and deliver content relevant to the user in the form of recommendations based on the information available about the user's preferences and interests, and the considered domain [40]. Using this information, RS attempt to predict the rating or preference the user would give to each of the unseen items under consideration and recommend those for which the prediction is the highest. High-performance recommender systems can be invaluable to online content providers to increase user satisfaction as content that better matches individual user preferences can be recommended. For advertisement-driven or pay-per-view businesses, this can boost revenues substantially by increasing viewing time and clicks, and for subscription-based businesses, the increased satisfaction can lead to higher popularity and customer loyalty. Being widely used in areas such as movies, news, articles, and e-commerce, recommender systems have become increasingly relevant.

Different approaches to recommender systems [38] exist based on the data they use and underlying assumptions applied: *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 options known as *hybrid RS* [8]. While the main assumption of collaborative filtering is that if two users have similar opinions on a particular issue, they will likely have a similar opinion on another issue, content-based filtering assumes that users like items that have similar contents, independent of the opinions of others, and thereby recommend to user unseen items that are the most similar to the items in the user profile, typically represented in the form of a vector of item features.

In this paper, we focus on content-based RS [36] operating on similarities between content items

based on various extractable features. The features available depend on the item type and dataset. For instance, typically news articles are attributed with an author, publication date, subject, and the full article content, whilst movies have a director, cast, runtime, release date, genre, and plot. Although text is the common form of information to extract features from to measure the similarity of items, other types of information (e.g., music songs include the artist, genre, and lyrics, movies include the actors, plot, and posters) can also serve as a source of features.

A widely used technique to estimate the similarity between texts is Term Frequency - Inverse Document Frequency (TF-IDF) [26], 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 [22]. Several recommenders such as Synset Frequency-Inverse Document Frequency (SF-IDF) [9], and its extension SF-IDF+ [33], Concept Frequency-Inverse Document Frequency (CF-IDF) [22], and its extension CF-IDF+ [16] have taken the TF-IDF concept further to provide recommendations of news articles, using synsets (S) from semantic lexicons or concepts (C) from domain ontologies for features instead of terms. These methods have further been extended to Bing-SF-IDF+ [10], Bing-CF-IDF+ [7], and Bing-CSF-IDF+ [51] recommenders by including semantically related synsets or concepts, and absorbing named-entity similarities using Bing page counts. In these recommender systems, a vector of weights is used to optimize the relative importance of the different features in the calculation of the similarities. The developments in these recommender systems have shown substantial improvements in performance.

Relying on the promising results of the aforementioned semantics-driven recommender systems for news articles, and encouraged by the successful scaling and porting of these methods to large-scale recommendations [5], we go a step further and explore the value of semantic information extracted from items more complex than text, namely digital images, derived by the idea that *a picture may be worth more than a thousand words*! In this paper, we continue our previous work on semantics-driven recommenders [5] and extend the extraction of semantic features from text to digital images (i.e., movie posters), and explore whether and to what extent multimodal content (both text and visual information) contributes to recommendations made. In particular, we seek to answer the following research questions:

RQ1: *Can the semantics-driven recommender systems, proven for news domain, be applied to a*

large-scale movie recommendation problem, where diverse information of different nature is available?

RQ2: *How to extract visual-semantic features from information such as digital images for recommendations?*

RQ3: *How do visual-semantic features extracted from digital images contribute to recommendations made?*

Multimodal learning is a hot topic nowadays being one of the tools to address the data bottleneck problem (lack of training data) [44]. Exploring text and visual features for modelling user profiles has been successfully exploited in recommender systems, for example, for restaurant recommendations [12], movie or restaurant recommendations with multimodal knowledge graphs [47], or using tweets with images to recommend hashtags [60]. Our research differs from these previous works as it uses more advanced, semantics-driven, approaches for both textual features and visual features for modelling the user and item profiles. To our knowledge, the combination of text, concepts, and images in the absence of an external knowledge graph is also unique in our solution.

The main contributions of this paper are as follows:

- A method for extracting visual-semantic features from digital images using computer vision for the task.
- An adjustment of the scaled similarity model [5] for visual-semantic features extracted from images.
- A proposal of a novel and unique method for large-scale semantics-driven recommendations based on concepts and synsets extracted from text, and synsets extracted from digital images, in the absence of an external knowledge graph.
- Demonstration that semantics-driven recommender systems, previously proven effective for text-based recommendations (e.g., the news domain), can be effectively utilized for multimodal visual-semantic recommendations with the proposed approach for various domains with diverse information available.

In this paper, we extend our previous work [6]. We expand the related works section, describe in more detail feature extraction from textual content and from visual content, provide a short

overview of the method of establishing a virtual domain ontology [5] based on existing data, and develop a deeper and broader discussion on recommendation method, experiments, evaluation, and results.

The rest of this paper is organized as follows. In Section 2 we discuss the related work. In the following Section 3, we address the data used for the research, and in Section 4 describe the recommendation methodology. Section 5 discusses the evaluation of the recommendation methodology and delivers the results of the experiments. Last, Section 6 draws conclusions and presents some ideas for future research.

2. Related Work

Combining text and visual features for a recommendation of items that have multimodal content to address the data bottleneck problem [44] has been addressed to a limited extent in previous literature [12, 47, 60]. Chu and Tsai [12] explored the use of text and visual features (images taken by customers) for restaurant recommendations. Through experiments, they verified visual information to aid favourite restaurant predictions. Zhang et al. [60] investigated the use of images in tweets, as a source of additional information, to recommend hashtags and exploited convolutional neural networks (CNNs) to extract features from images and long short-term memory networks (LSTM) to extract features from the text. Their experiments with Twitter tweets showed the inclusion of images to provide better recommendation performance over the use of textual information only. Sun et al. [47] use the ResNet50 [25] models to extract visual features from movie frames and dish images, Smooth Inverse Frequency (SIF) [3] for text embeddings, and learn concept embeddings from a knowledge graph for movie (interestingly, and similar to our study, they also use the *Toy Story* movie example) and restaurant recommendations. In comparison to our work, these methods use low-level representations difficult to interpret compared to the synset-based representations in our method. Also, we use a virtual ontology that can be easily built from any semi-structured data available and do not assume the existence of an a priori knowledge graph like required in [47]. None of the existing solutions make use of the high-level visual features based on synsets and visual-semantic embeddings that fuse text and image information in one embedding.

The most used technique to represent text features is TF-IDF [26]. Although, alternative techniques like SIF [3] and BERT [18] to embed text exist, our purpose here is not to use the

most advanced text representation technique but to check if the visual and conceptual features can help make better predictions in a context where a knowledge graph is non-existing. Thereby, we continue by reviewing the semantics-driven recommenders TF-IDF, CF-IDF, SF-IDF, and their extensions CF-IDF+ and SF-IDF+ originally designed for news recommendation. For the latter reason, the performance evaluations in the literature are reported on a dataset of news items. The dataset used to compare the recommenders consists of user profiles indicating interest/disinterest towards each seen news item. Unseen news items for which the normalized similarity prediction is higher than a predetermined threshold value are recommended. The performance is evaluated through the widely used F_1-measure. Although these recommender systems extract features from the text of news articles, they can be used to predict similarity between any two texts.

The Term Frequency-Inverse Document Frequency (TF-IDF) is of interest as SF-IDF(+) and CF-IDF(+) recommenders build on its mathematical concept [26]. The TF-IDF [43] 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), is constant over all documents and can be seen as a weight that gives relative importance to rare terms. Before counting terms, a pre-processing step is performed to remove noise and increase performance. Stop words are removed and all other words are lemmatized so that all words with the same root are considered to be the same term [22]. The resulting feature vector represents terms with scores, which can be compared to user vectors using similarity functions (e.g., the cosine similarity). The TF-IDF score is large for terms that occur frequently in a 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) [22] recommender system 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 [22]. CF-IDF+ extends this method further by including directly related concepts in the domain ontology [16]. 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 inclusion of related concepts can add more domain semantics to the feature vector.

The Synset Frequency-Inverse Document Frequency (SF-IDF) [9] is another variant of TF-IDF, which in addition to all terms looks at synonyms and ambiguous terms using a semantic lexicon (WordNet). This generally results in a longer vector than CF-IDF because more matches are found as WordNet is much larger than a typical domain ontology. 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. Similarly to CF-IDF+, SF-IDF+ [33] extends SF-IDF and includes 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, outperforming SF-IDF.

Bing-SF-IDF+ [10] is an extension of SF-IDF+, which in addition to words in the semantic lexicon also considers the similarity between named entities frequently occurring on the Web through a separate similarity measure – Bing distance, based on the number of page counts originating from the Bing search engine. This measure is a function of three search result page counts: two counts for each entity separately and one for a combination of the two. An optimized weight is used to combine the Bing distance and the SF-IDF+ cosine similarity, leading to improved performance over SF-IDF+ [10]. Similarly, Bing-CF-IDF+ [7] and Bing-CSF-IDF+ [51] advantage from the inclusion of named entities and their similarity, while Bing-CF-IDF+ outperforms CF-IDF+, and Bing-CSF-IDF+ outperforms Bing-SF-IDF+ and CF-IDF+. In this research, we do not use Bing distance evaluations, as the named entities present in the plots are generally names of fictional characters and would only rarely provide substantial information. Although Bing distance metrics could be applied to the persons involved with the movie (i.e., directors, actors, and writers), they will be in our research represented as concepts in the domain ontology. Considering also the fact that a large number of named entities makes the pair-wise search queries infeasible, we decide to exclude Bing distances from our recommender system in this research.

The previously discussed TF/CF/SF-IDF(+) content-based RS were originally established for

news recommendation, a 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 [5] on the example of the 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, removing 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 [5]. While [5] used semantic information available in the textual form for large-scale recommendations on the example of the movie domain, this work extends the recommendation problem further by including also rich semantic information available in graphical form on movie posters (digital images).

Recommender systems for the media and multimedia domain are of interest to many researchers due to the enormous amount of diverse information available. Various approaches have been exercised to provide recommendations: a graphical model and signature-tree-based scheme over social media streams [62], knowledge graphs [24], context-aware social media recommendations [61], and ontologies [2, 45], Bidirectional Encoder Representations from Transformers (BERT) [18] for conversational RS [37] with experiments on movies, books, and music recommendation, Word2Vec algorithm to recommend movies [57] based on metadata (e.g., directors, and actors), and textual image metadata for recommending socially relevant images [27]. A comprehensive overview of RS for multimedia content is given in [17].

Computer vision uses algorithms to gain a high-level understanding of visual information, e.g., digital images. 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) [13, 14], facial recognition [29, 32], face detection [21], or to learn image shapes for recommending apparel goods [42]. On some object classification tasks [48] it can even rival human performance [41]. CNNs are types of feed-forward artificial neural networks (ANNs) [54], which are models inspired by the connectivity patterns of neurons (called nodes in ANNs) in the animal and human brains.

Guo et al. [23] used CNNs to extract features of semantic image objects, splitting the image into a number of image objects, extracting the features, and then summarising the results for an image.

Tuinhof et al. [49] 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. [58] on the other hand focused on recommending goods based on image content represented by a weighted feature model using only computationally inexpensive low-level image features such as colour, texture, and shape to cut down on computation time. The 19-layers deep trained convolutional neural network VGG19 has proven itself for large-scale image classification task [46], being successfully applied for a wide range of various tasks from medical image processing [1, 19], detection of computer-generated realistic-looking images [11], post-disaster damage assessment [34], to detection of facial expression behind masks [56]. VGG19 has also been used to establish visual-semantic embeddings (VSE) [28] used for the challenge of image captioning [53] to generate a natural language caption that best describes the content of an input image. In this research, we use computer vision to extract visual-semantic information from movie posters.

Considering the aforementioned recommender systems, previously used for small-scale news article recommendations, where the recommenders operated on text only, in this research we want to explore other types of items, for which text might only describe some aspect of the item. For example, music songs might include a description of the artist, the album, or the lyrics. Movies can have their plots, storylines, posters, or other descriptions available. Thereby, the extraction of semantic features from information of a different nature, and their contribution to semantics-driven recommender systems is worth exploring. We specifically explore a domain that is substantially different from news articles – movie recommendations – for which information on thousands of movies and millions of user ratings is available, and focus on recommending items of multimodal content by combining textual and visual information.

3. Recommendation Data

The problem that we focus on in this research is the large-scale recommendation of movies. As in our previous work [5], where we addressed the scalability of the semantics-driven recommenders on content represented by text, we continue to use the MovieLens 20M[1] dataset from the GroupLens Research Project[2] at the University of Minnesota. The MovieLens dataset provides 20,000,000 user

[1] https://grouplens.org/datasets/movielens/20m/

[2] https://grouplens.org/

ratings on a scale of 1–5 for 27,278 movies over a ten-year period (9 January 1995 until 31 March 2015) created by 138,493 users who had rated at least 20 movies. We acquire from the MovieLens[3] the title, year of release, genre labels, and IMDB[4] identification numbers (ids) for each movie as the item-level information for feature extraction.

We use two other sources to collect movie information over IMDB ids: (i) OMDb[5], which freely provides an API in the form of a RESTful Web service, to query movie plots, and (ii) TMDb[6] to collect movie posters over its provided API. 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.

The combined data contains many movie-level variables out of which for this research we choose to retain only those containing substantial semantic information, as those could be valuable for semantics-driven recommendations. We believe the bulk of this semantic information is represented by the names of persons involved in the movie, the genre(s), the plot, and the poster. The involved persons are the actor(s), director(s), and writer(s). 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 together with their descriptive statistics.

4. Recommendation Methodology

This section covers the extraction of semantic features from the plots and digital images (movie posters). We shortly describe how we find related concepts without the need for an external

[3]https://movielens.org/

[4]The Internet Movie Database, https://www.imdb.com/

[5]The Open Movie Database, https://www.omdbapi.com/

[6]The Movie Database, https://www.themoviedb.org/

Table 1: Movie-level variables and their sources used in the research together with descriptive statistics (N - number of data items in source).

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.

ontology, and 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 [22], CF-IDF(+) [16, 22], and SF-IDF(+) [10] 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 [5]. To extract terms and synsets from the plots, we use NLP techniques that can filter out noise from the plots and exploit known regularities in natural language. Using the NLTK[7] package in Python 2.7, each plot is split into a set of sentences and 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) and separated by spaces or commas. Using POS tagger, each word is tagged with its POS (i.e., noun, verb, adjective, and adverb). Stop words that contain negligible semantic information (e.g., *the*, *is*, and *at*) are then removed. Next, we apply the Porter [52] stemming algorithm to each word to reduce the words to their roots and extract the terms. For instance, $fisher$, $fishing$,

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

and $fished$ are reduced to the same root $fish$. This way words with a similar basic meaning are considered to be the same term.

Synsets are extracted using the Adapted Lesk [4] WSD-algorithm, an improved [35] version of the classic Lesk [30] algorithm, on each word. WSD addresses the problem of identifying the sense of a word – the meaning in its context. For example, the noun *bank* has multiple meanings that are very different and the meaning in a text has to be judged in the context in which it is used. Adjusted Lesk does this by calculating a similarity between the context (sentence) of the word in the text and the definition of each sense of the word from the dictionary (in our case WordNet). The sense with the highest similarity is then identified. 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. Domain Ontology

Domain ontologies are considered resources that are external to the dataset from which the concepts are derived, and subsequently have to either be obtained through external sources or manually constructed specifically for the purpose of the recommender system. In [5] we proposed a general method as an alternative to external domain ontologies, solely based on the dataset itself, which allows to find concepts related through a common item by a series of matrix multiplications of binary matrices.

We apply the proposed method [5] on our dataset of 25,138 movies, which contains 12,231 directors, 45,393 actors, 27,415 writers, 19 genres from the MovieLens and 27 genres from the OMDb, and consider *actors*, *directors*, *writers*, and *genres* as concept classes. Based on the average number of these concepts per movie (Table 1), we estimate that there are 292,587 bidirectional movie-concept relations that implicitly form a virtual domain ontology, which can be used to find related concepts based on item-concept occurrences. Through several matrix multiplications of these feature matrices, we obtain a matrix of related concepts, which allows us to find related concepts through their relations. In a simplified form this can be represented as:

$$item_u \xrightarrow{contains} concept_{a(i)} \xrightarrow{occurs\ in} item_v \xrightarrow{contains} concept_{b(j)}, \tag{1}$$

where items refer to movies, and a, b are concept classes.

For example, the method allows to find the related directors of the movies through the movies' actors. This saves us from manually constructing domain ontology for movies.

4.3. Feature Extraction from Images

Let us now consider semantic feature extraction from digital images, for which we apply computer vision algorithms to gain high-level understanding of visual information on digital images. In particular, we use Convolutional Neural Networks (CNNs) [20] which are state-of-the-art models in computer vision to extract a vector of synset probabilities and a Visual-Semantic Embedding (VSE) vector from each movie poster.

In a digital image, each pixel 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[8]) 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[9] take floating-point numbers as inputs, usually single precision floats (32 bits), the matrix is normalized by multiplying with $\frac{1}{255}$ to obtain a matrix of values in the range $[0, 1]$.

In this research, we use movie posters in the form of digital images to extract semantic features. Movie 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 plots, the posters contain fewer irrelevant elements. We now continue by describing the extraction of synset vectors and visual-semantic embeddings.

4.3.1. Synset Vectors

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 [46]. VGG19 was the highest-performing submission for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[10]

[8] http://opencv.org/

[9] http://deeplearning.net/software/theano/

[10] http://www.image-net.org/challenges/LSVRC/

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% [41]. The trained parameters for this model are publicly available[11].

VGG19's convolutional layers each have a filter size of 3×3 and the input to each of those layers is zero-padded with a border of one pixel 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 pixels 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. Figure 1 exemplifies these windows on the poster for the movie *Toy Story* with identified synsets and their probabilities shown on the left of the poster image. 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 semantic feature vectors of 1,000 synset values (class predictions mapping to a vocabulary). Synsets, even with the smallest probability (e.g., "Tobacco shop" on Fig. 1), are part of the profile (one of the 1000 entries in the VGG19 features in the user profile given by Fig. 2) and are treated the same as any other feature in the user profile.

4.3.2. Visual-Semantic Embeddings

The 1,000 synset values as class predictions 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) [28] that has been used for the challenge of image captioning [53], 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

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

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

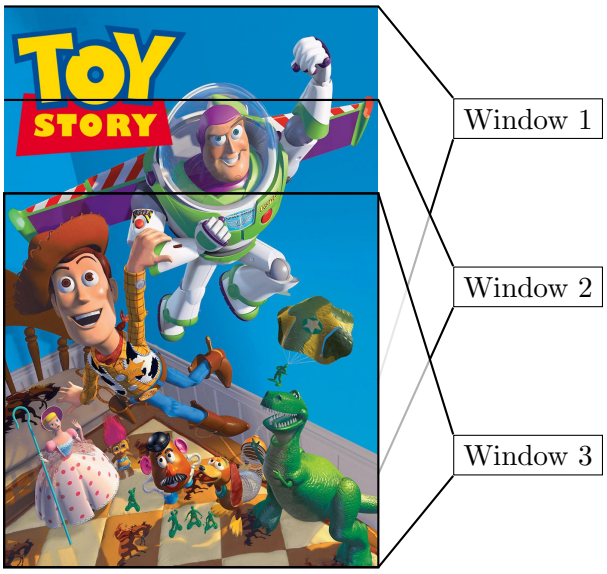

Figure 1: Crop of three windows of 224 × 224 px over the movie poster (right), and the predicted class (synset) probabilities for the given movie poster (left). Feature values are the maximum probabilities of each synset.

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 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 [28] by a form of transfer learning [55], 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:

$$\begin{aligned} &\sum_m \sum_r max\{0, \alpha - s(\vec{m}, \vec{c}) + s(\vec{m}, \vec{c_r})\} \\ +&\sum_c \sum_r max\{0, \alpha - s(\vec{c}, \vec{m}) + s(\vec{c}, \vec{m_r})\} \end{aligned} \tag{2}$$

where $s(\vec{m}, \vec{c}) = \vec{m} \cdot \vec{c}$ is the scoring function. As [28], 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 [28] have made an embedding matrix to generate 1,024-dimensional visual-semantic embeddings publicly available[12]. This matrix was trained to optimize Eq. 2 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 a disadvantage – the features are hidden and have no natural interpretation, making it complicated to link them to an ontology or semantic lexicon.

4.4. Scaling Visual Features

When all features are extracted, we have to consider how to scale them. The traditional method of scaling of TF-IDF is the Inverse Document Frequency (IDF), which we apply to terms and synsets from the plots, in line with SF-IDF(+). For concepts, the scaling is different from

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

CF-IDF+. As we only extract occurrences of concepts, which are always in $\{0, 1\}$, we do not apply IDF scaling, as it would deviate from its original meaning (relative frequency).

The 1,000 synset values (VGG19) and the 1,024 VSE values extracted from the movie posters could also 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, which are hidden and do not have a natural interpretation. 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). These learned scales are the weights related to the 1000 VGG19 features and 1024 VSE features. We apply the established similarity model scaling [5] also to synsets and visual-semantic features extracted from the 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. For finding related concepts, we limit ourselves to single-step paths of directors, actors, and writers, i.e., $m_1 = m_2 = m_3 = 1 + 3 = 4$, for the genres $m_4 = m_5 = 1$, and as there are no relations among terms or visual-semantic features $m_6 = m_8 = m_9 = 1$. Since we retrieve 18 relations from WordNet and only for the plot synsets, $m_7 = 1 + 18 = 19$ and $m_8 = 1$. Figure 2 visualizes the item/user model vector of a user profile. One can see that it contains all feature types listed in Table 2, and is based on $205,169$ features.

To learn scaling for visual feature types, we use the similarity model (Eq. 3) established in [5], where s_i is part similarity (here cosine similarity) and w_i the weight of part similarity s_i (from loss function), $\vec{u_i}$ user-profile feature vector, $\vec{v_i}$ unseen item feature vector (sum of the vectors associated to previously clicked items), $\vec{q_i}$ vector of relation weights, $\vec{U_i}$ user feature matrix, and $\vec{V_i}$ feature matrix for unseen items:

$$sim = \sum_{i=1}^{k} w_i s_i = \sum_{i=1}^{k} w_i \cdot cos(\vec{u_i}, \vec{v_i}) = \sum_{i=1}^{k} w_i \frac{\vec{q_i}(U_i V_i^\top)\vec{q_i}^\top}{\sqrt{\vec{q_i}(U_i U_i^\top)\vec{q_i}^\top}\sqrt{\vec{q_i}(V_i V_i^\top)\vec{q_i}^\top}} \tag{3}$$

Table 2: Item/user model feature types and their characterization by type, source dataset, the number of features n_i, and the number of relations m_i between features of type i.

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.

In the similarity model (Eq. 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. The part-similarity model s_i changes to Eq. 4, where $\vec{u_i} = U_i$, and $\vec{v_i} = V_i$, because the number of relations $m_i = 1$ for these feature types:

$$s_i = \frac{\vec{q_i}((\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{v_i})^\top)\vec{q_i}^\top}{\sqrt{\vec{q_i}((\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{u_i})^\top)\vec{q_i}^\top}\sqrt{\vec{q_i}((\vec{c_i} \circ \vec{v_i})(\vec{c_i} \circ \vec{v_i})^\top)\vec{q_i}^\top}} \tag{4}$$

We restrict $\sum_{l=1}^{m_i} \vec{q}_{il} = 1$ and here $m_i = 1$, making $\vec{q_i} = 1$ and redundant (Eq. 5):

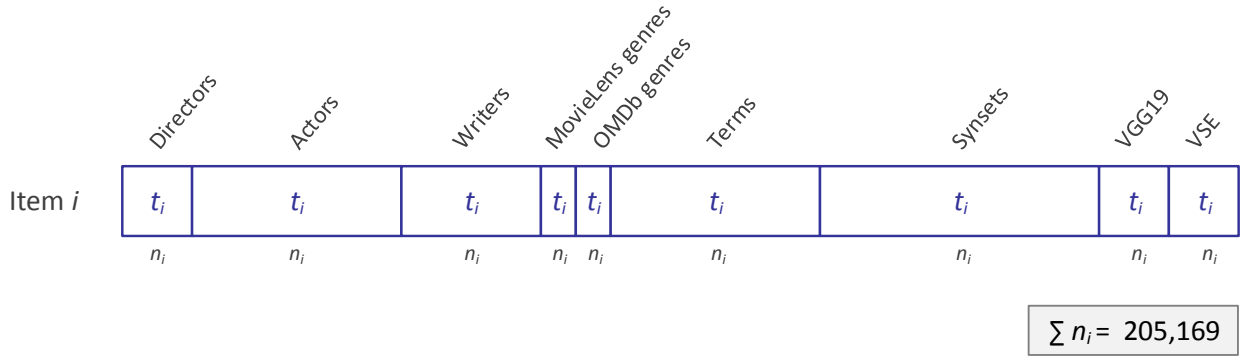

Figure 2: Visualization of the item/user model vector of a user profile.

$$sim = \sum_{i=1}^{k} w_i s_i = \sum_{i=1}^{k} w_i \frac{(\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{v_i})^\top}{\sqrt{(\vec{c_i} \circ \vec{u_i})(\vec{c_i} \circ \vec{u_i})^\top}\sqrt{(\vec{c_i} \circ \vec{v_i})(\vec{c_i} \circ \vec{v_i})^\top}} \tag{5}$$

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 [5] with pre-computed $U_i U_i^\top$, $U_i V_i^\top$, and $V_i V_i^\top$.

5. Experiments and Results

We train the similarity model directly 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 [5]. To optimize (train) the part-similarity weights $\vec{w}$ and the relationship weights $\vec{q_i}$, we apply stochastic gradient descent (SGD) on the gradient of the similarity model. The target similarity $y \in \{0, 1\}$ is defined as $y = I[user\ likes\ item]$. The evaluation consists of calculating various classification metrics between the test users' actual likes/dislikes versus the recommendations.

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 feature matrices $U_i V_i^\top$, $U_i U_i^\top$, and $V_i V_i^\top$ (where U_i denotes user feature matrix and V_i feature matrix for unseen items) are constructed from the pre-computed data matrices X_i, the result of matrix multiplications. 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 present in the user profile. For each user profile, we sample a liked or disliked item with an 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 recommender system 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 be 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 similarity model is trained with SGD and follows the method (Algorithm 1) described in [5].

We evaluate our method on the sampled 809 test user profiles using the trained model to predict the similarity score for each unseen item in a batch. The comparison between the predicted scores and the actual likes forms the basis of performance measurement. For each threshold value $\tau \in \{\frac{i}{500} \ \forall i \in (0, 500)\}$ the unseen items for which $sim > \tau$ are recommended. We can define $\hat{y} = I[sim > \tau] \in \{0, 1\}$ to indicate this. From these recommendations, we can obtain the number of true positives $TP = \sum \hat{y} \cdot y$, false positives $FP = \sum \hat{y}(1-y)$, true negatives $TN = \sum (1-\hat{y})(1-y)$, and false negatives $FN = \sum y(1-\hat{y})$ for each user. This allows us to calculate *precision* (Eq. 6) as a measure of what fraction of recommended items were actually liked by users, and *recall* (Eq. 7) as a measure of how many liked items were recommended.

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

We further calculate *specificity* (Eq. 8) to measure how many disliked items were not recommended, and another widely used measure for classification performance – *accuracy* (Eq. 9), which measures the fraction of correct judgements.

$$Specificity = \frac{TN}{TN + FP} \tag{8}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{9}$$

Note that precision can be increased by recommending as few items as possible, in this case by using a high threshold (until it is undefined for $TP + FP = 0$). On the other hand, recall can be increased by lowering the threshold. In fact, a perfect score of $Recall = 1$ can be obtained by simply recommending all items. Therefore, given some similarity predictions, there is a trade-off between precision and recall. This trade-off can be explored by varying the threshold, and plotted in a precision-recall (PR) curve. The area under this curve, $AUC(PR) \in [0, 1]$, is a measure of recommender quality that takes this trade-off into account. There also exists a trade-off between recall and specificity, which is illustrated by the receiver operating characteristic (ROC) curve – a plot of $Recall$ versus $(1 - Specificity)$. Similar to $AUC(PR)$, we calculate $AUC(ROC)$ as a measure of recommender quality (Table 4).

Another metric that uses both precision and recall is the F_1-measure (Eq. 10), also considered with a trade-off for maximizing the measure. If the goal is to maximize $F_1 \in [0, 1]$, a recommender system has to estimate through some model $\tau = r(...)$ the optimal threshold τ to determine how many items to recommend. These estimations are beyond the scope of our research so we report (Table 4) F_1 under two assumed models for r. If r can only find an optimal threshold that is static across users, we consider this the worst case, denoted by $\min_r(F_1)$. As an upper limit of F_1, we additionally measure $\max_r(F_1)$, which assumes r can estimate the optimal threshold for each user perfectly.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{10}$$

The accuracy measure $Accuracy \in [0, 1]$ does not take into account that even uninformative recommenders can achieve high accuracy, depending on the distribution of y, so we correct for this by calculating Cohen's kappa [15] coefficient κ (Eq. 11), where the expected accuracy is found as given by Eq. 12. This takes the expected accuracy into account and provides a more meaningful measure of classification power.

$$\kappa = \frac{Accuracy - ExpectedAccuracy}{1 - ExpectedAccuracy} \tag{11}$$

$$ExceptedAccuracy = \frac{(TN+FP)(TN+FN)+(FN+TP)(FP+TP)}{(TP+FP+TN+FN)^2} \quad (12)$$

Like the F_1-measure, κ also varies by threshold, so we again calculate $\min_r(\kappa)$ and $\max_r(\kappa)$ under assumptions about the ability of the model $\tau = r(...)$ to predict the optimal threshold τ that maximizes κ.

For each of the performance metrics, we calculate the average over the 809 test users. We use these 809 user profiles (a user profile is just the sum of the vectors that represent the liked item (movies) features) in the test set to measure the performance of the models. The experiments are repeated with various feature types included or excluded to measure their contribution to the model's performance.

We demonstrate the value of inclusion of visual features to semantics-driven recommendations by comparison to the traditional TF-IDF recommender (further 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 the genres from MovieLens and OMDb) and operating on the ontology as C, VGG19 synsets (unscaled) as VG, and VSE (unscaled) 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 $\mathrm{VG_L}$ or $\mathrm{VS_L}$, respectively. When the VG scaling is pre-trained in another model and transferred to this model, we denote the component $\mathrm{VG_R}$ (each of the 10 restarts uses a pre-trained scaling from a different restart of $\mathrm{VG_L}$) or $\mathrm{VG_A}$ (each of the 10 restarts uses the same pre-trained scaling – the average scaling over all 10 restarts of $\mathrm{VG_L}$). Our proposed semantics-driven model is called C+S+$\mathrm{VG_A}$, 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 $\mathrm{VG_L}$ models. Table 3 lists all twelve models used. We test the proposed C+S+$\mathrm{VG_A}$ model against the TF-IDF benchmark, and against all alternative models.

Let us start by describing the results for the computational load of the optimization procedure implemented in Python 2.7 [13] using Keras[14], which uses a back-end supported by Theano[15]. The calculations are performed on a regular desktop PC with NVIDIA GTX1060 CPU enabling efficient parallel computations of the gradient updates in batches of 1,024 observations. The gradients are

[13] https://www.python.org/download/releases/2.7/

[14] (https://keras.io)

[15] https://pypi.org/project/Theano/

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****		
			Validation	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.

calculated automatically by Theano using backpropagation. We minimize the overhead of online loading by using a solid-state drive (SSD) and simultaneously loading the next batch while SGD is applied to the current batch. 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 the training time 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 15-times reduction in seconds per epoch of the VG model, which uses pre-computed dot-products, compared to its VG_L counterpart 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 1.5-times as many epochs to converge and results in a slightly better logloss. The sparsity of the VGG19 vectors compared

Table 4: Performance metrics on test set, $n = 809$ user profiles, averages over 10 random restarts (min_r – worst case with static threshold across users, max_r – upper limit with optimal threshold for each user).

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

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 AUC for the PR and ROC curve, F_1-measure, and Cohen's kappa [15] coefficient κ. Even though we do not directly optimize for these metrics, a lower logloss results in higher test performance (Table 4).

The analysis of performance metrics over all models (Table 4) shows that concepts alone (C) are more informative than both synsets (S) and terms (T), as C substantially improves over the baseline T on all metrics, and the combination of C+S [5] outperforms T on all metrics, regardless the unexpectedly poor performance of S, likely caused by the low-quality WSD. A look at the relation weights of concepts (Table 5) confirms that for all models the directly found concepts have the highest average optimal weight. We can also infer from the maximum weights that there are a few optimal solutions in which one of the other relations holds most of the weight. For the indirect

concepts of the *actors* class, the most dominant relation is *writers*, suggesting that users tend to value movies with actors who have played a role in movies that involve writers from their user profile. Related writers on the other hand receive little weight overall, regardless of the relation through which they are found.

The inclusion of features captured from poster images further improves (depending on the method) the recommendation as the proposed C+S+VG_A model outperforms C+S and thereby also the benchmark TF-IDF (T) (Table 4). 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.

In contrast to the visual-semantic features, the synsets are interpretable and we can compare the 1,000 learned scales. We optimize the same scaling in a full C+S+VG_L model as a benchmark to compare C+S+VG_A and C+S+VG_R against and estimate the impact of re-using or transferring learned scales across models. The top 10 synsets with the highest scale, on average over 10 restarts (Table 6) exhibit some consistency. The average correlation for VG_L of the visual scaling over the 10 restarts is 0.268 ($n = 45$), while for the C+S+VG_L it is higher at 0.486 ($n = 45$). Due to the much higher dimensional space compared to the relation weights, less stability can be expected. Nevertheless, the correlations indicate that there is some stability across solutions, and this increases when concepts and synsets are added.

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 (Table 4). The performance is indistinguishable from that of C+S+VG_L, which can be considered a good benchmark for the learned scaling because this model optimized the scaling together with the rest of the models. 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 (Table 4). Average AUC(ROC) improves from 0.535 to 0.634, and

Table 5: Optimized concept relation weights, means and maxima over 10 random restarts. $\xleftarrow{relation}$ denotes found through *relation*.

		C	C+ S	C+ S+ VG	C+ S+ VG_L	C+ S+ VG_A	C+ S+ VG_R
Mean	Actors	.268	.774	.772	.271	.932	.360
	$\xleftarrow{Actors}$	.000	.000	.000	.272	.000	.174
	$\xleftarrow{Directors}$	.000	.000	.000	.272	.000	.174
	$\xleftarrow{Writers}$	.700	.011	.152	.417	.003	.398
	Directors	.697	.643	.666	.741	.602	.620
	$\xleftarrow{Actors}$	.002	.000	.002	.005	.001	.005
	$\xleftarrow{Directors}$	.272	.318	.287	.239	.338	.330
	$\xleftarrow{Writers}$	.030	.039	.046	.016	.059	.045
	Writers	.880	.886	.764	.888	.898	.888
	$\xleftarrow{Actors}$	.048	.037	.030	.045	.023	.045
	$\xleftarrow{Directors}$	.065	.054	.060	.054	.045	.056
	$\xleftarrow{Writers}$	.008	.023	.147	.013	.034	.012
Max	Actors	.911	.909	.907	.923	.942	.936
	$\xleftarrow{Actors}$	.000	.000	.001	.965	.001	.654
	$\xleftarrow{Directors}$	.096	.969	.094	.079	.095	.225
	$\xleftarrow{Writers}$	.985	.019	.971	.977	.005	.980
	Directors	.710	.685	.721	.762	.704	.834
	$\xleftarrow{Actors}$	.004	.001	.014	.008	.006	.016
	$\xleftarrow{Directors}$	.298	.394	.390	.335	.578	.712
	$\xleftarrow{Writers}$	.107	.101	.164	.102	.384	.276
	Writers	.912	.897	.920	.923	.908	.903
	$\xleftarrow{Actors}$	.057	.056	.055	.054	.026	.081
	$\xleftarrow{Directors}$	.079	.060	.077	.067	.073	.081
	$\xleftarrow{Writers}$	.031	.038	.907	.029	.044	.043

Table 6: Learned visual feature scalings (weights), $n = 809$ users. Top 10 synsets with highest scales, mean over 10 restarts.

VG_L		C+S+VG_L	
Fur coat	.01198	Book jacket	.01587
Stage	.00991	Toyshop	.01144
Web site	.00975	Bow tie	.01142
Balloon	.00935	Web site	.01142
Jigsaw puzzle	.00930	Jigsaw puzzle	.01000
Volcano	.00923	Volcano	.00973
Pick	.00907	Cinema	.00911
Toyshop	.00869	Sweatshirt	.00887
Cinema	.00861	Fountain	.00886
Bow tie	.00855	Military uniform	.00847

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 0.041 to 0.137 and from 0.200 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 research, we continued our work on semantics-driven recommender systems and demonstrated that these systems are broadly applicable beyond news recommendations. In particular, we continued our work on scaling content-based semantics-driven recommenders to a large-scale recommendation task and extended the approach to include visual-semantic features delivered by computer vision. The paper delivers the second phase of our work [5], where we previously 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 a 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. In this paper, 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 recommender systems can be extended to more complex domains than news recommendations with high-quality recommendations on an extremely large scale multimodal content.

The proposed visual-semantic recommender C+S+$\mathrm{VG_A}$ with visual features extracted from digital images by means of computer vision strongly outperformed the baseline model 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 recommenders have many unexplored applications and can be utilized effectively with the proposed approach to various domains. 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.

As for the ideas for further research, one potential direction would be to use another kind of text embedder, e.g., BERT [18] for building textual features instead of TF-IDF approach, and visual embedder, e.g., Swin Transformer [31] for building additional visual features for the model. Presently, related synsets and concepts are found through direct connections but multi-hops instead of single hops when navigating through WordNet or domain ontology could add additional semantics. The benefits of allowing a concept to be in multiple classes, such as a person who is a director of one movie and an actor in another, are also left to explore. It would also be worth investigating the effect of using DBpedia to exploit more relations than MovieLens provided. And lastly, we would like to apply model regularization by penalizing the learned weights and further increase model capacity.

References

[1] M. J. Ahmed and P. Nayak. Detection of lymphoblastic leukemia using VGG19 model. In *2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pages 716–723, 2021. doi: 10.1109/I-SMAC52330.2021.9640955.

[2] M. Arafeh, P. Ceravolo, A. Mourad, E. Damiani, and E. Bellini. Ontology based recommender system using social network data. *Future Generation Computer Systems*, 115:769–779, 2021. ISSN 0167-739X.

[3] S. Arora, Y. Liang, and T. Ma. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations*, 2017.

[4] S. Banerjee and T. Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. In A. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, (CICLing '02), pages 136–145. Springer, 2002.

[5] M. M. Bendouch, F. Frasincar, and T. Robal. Addressing scalability issues in semantics-driven recommender systems. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI-IAT '21)*, New York, NY, USA, 2021. ACM. doi: 10.1145/3486622.3493963.

[6] M. M. Bendouch, F. Frasincar, and T. Robal. Enhancing semantics-driven recommender systems with visual features. In *Advanced Information Systems Engineering: 34th International Conference (CAiSE 2022)*, pages 443–459. Springer International Publishing, 2022. ISBN 978-3-031-07472-1. doi: 10.1007/978-3-031-07472-1_26.

[7] E. Brocken, A. Hartveld, E. de Koning, T. van Noort, F. Hogenboom, F. Frasincar, and T. Robal. Bing-CF-IDF+: A semantics-driven news recommender system. In P. Giorgini and B. Weber, editors, *Advanced Information Systems Engineering*, pages 32–47, Cham, 2019. Springer.

[8] R. Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[9] M. Capelle, F. Frasincar, M. Moerland, and F. Hogenboom. Semantics-based News Recommendation. In *Proc. of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-0915-8.

[10] M. Capelle, M. Moerland, F. Hogenboom, F. Frasincar, and D. Vandic. Bing-SF-IDF+: A Hybrid Semantics-Driven News Recommender. In *Proc. of the 2015 ACM Symposium on Applied Computing*, SAC '15, page 732–739, New York, NY, USA, 2015. ACM.

[11] T. Carvalho, E. R. S. de Rezende, M. T. P. Alves, F. K. C. Balieiro, and R. B. Sovat. Exposing computer generated images by eye's region classification via transfer learning of VGG19 CNN. In *16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 866–870, 2017. doi: 10.1109/ICMLA.2017.00-47.

[12] W.-T. Chu and Y.-L. Tsai. A hybrid recommendation system considering visual information for predicting favorite restaurants. *World Wide Web*, 20(6):1313–1331, nov 2017. ISSN 1386-145X. doi: 10.1007/s11280-017-0437-1.

[13] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *Proceedings of the 22nd International Joint Conference*

on Artificial Intelligence, IJCAI'11, pages 1237–1242. AAAI Press, 2011. ISBN 978-1-57735-514-4.

[14] D. Ciregan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, 2012. doi: 10.1109/CVPR.2012.6248110.

[15] J. Cohen. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1): 37–46, 1960.

[16] E. de Koning, F. Hogenboom, and F. Frasincar. News Recommendation with CF-IDF+. In *30th International Conference on Advanced Information Systems Engineering (CAiSE 2018)*, volume 10816 of *LNCS*, pages 170–184. Springer, 2018.

[17] Y. Deldjoo, M. Schedl, P. Cremonesi, and G. Pasi. Recommender systems leveraging multimedia content. *ACM Comput. Surv.*, 53(5):1–38, 2020. ISSN 0360-0300. doi: 10.1145/3407190.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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)*, pages 4171–4186, Minneapolis, MN, 2019. Association for Computational Linguistics.

[19] L. Devnath, S. Luo, P. Summons, and D. Wang. Performance comparison of deep learning models for black lung detection on chest x-ray radiographs. In *Proceedings of the 3rd International Conference on Software Engineering and Information Management*, ICSIM '20, page 150–154, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450376907. doi: 10.1145/3378936.3378968.

[20] M. Egmont-Petersen, D. de Ridder, and H. Handels. Image Processing with Neural Networks—a review. *Pattern Recognition*, 35(10):2279–2301, 2002. ISSN 0031-3203.

[21] S. S. Farfade, M. J. Saberian, and L.-J. Li. Multi-view face detection using deep convolutional neural networks. In *5th ACM on International Conference on Multimedia Retrieval*, ICMR '15, page 643–650, New York, NY, USA, 2015. ACM. ISBN 9781450332743.

[22] F. Goossen, W. IJntema, F. Frasincar, F. Hogenboom, and U. Kaymak. News Personalization Using the CF-IDF Semantic Recommender. In *Proceedings of the 1st International Conference on Web Intelligence, Mining and Semantics*, WIMS '11, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0148-0.

[23] G. Guo, Y. Meng, Y. Zhang, C. Han, and Y. Li. Visual semantic image recommendation. *IEEE Access*, 7: 33424–33433, 2019.

[24] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2020.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

[26] K. S. Jones. A Statistical Interpretation of Term Specificity and its Application in Retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

[27] R. Karlsen, N. Elahi, and A. Andersen. Personalized recommendation of socially relevant images. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, WIMS '18, New York, NY, USA, 2018. ACM. ISBN 9781450354899.

[28] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying Visual-Semantic Embeddings with Multimodal Neural Language models. *CoRR*, abs/1411.2539, 2014. URL `http://arxiv.org/abs/1411.2539`.

[29] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face Recognition: a Convolutional Neural-Network Approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997. ISSN 1045-9227.

[30] M. Lesk. Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1.

[31] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00986.

[32] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda. Subject Independent Facial Expression Recognition with Robust Face Detection using a Convolutional Neural Network. *Neural Networks*, 16(5–6):555 – 559, 2003. ISSN 0893-6080. Advances in Neural Networks Research: IJCNN '03.

[33] M. Moerland, F. Hogenboom, M. Capelle, and F. Frasincar. Semantics-based News Recommendation with SF-IDF+. In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*, WIMS '13, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1850-1.

[34] S. Nag, T. Pal, S. Basu, and S. Das Bit. Cnn based approach for post disaster damage assessment. In *Proceedings of the 21st International Conference on Distributed Computing and Networking*, ICDCN 2020, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450377515. doi: 10.1145/3369740.3372753.

[35] R. Navigli. Word Sense Disambiguation: A Survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, Feb. 2009. ISSN 0360-0300. doi: 10.1145/1459352.1459355.

[36] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The Adaptive Web: Methods and Strategies of Web Personalization*, volume 4321, pages 325–341. Springer Berlin Heidelberg, 2007. doi: 10.1007/978-3-540-72079-9_10.

[37] G. Penha and C. Hauff. What does BERT know about books, movies and music? Probing BERT for conversational recommendation. In *14th ACM Conference on Recommender Systems*, RecSys '20, page 388–397, New York, NY, USA, 2020. ACM. ISBN 9781450375832. doi: 10.1145/3383313.3412249.

[38] A. H. N. Rafsanjani, N. Salim, A. R. Aghdam, and K. B. Fard. Recommendation Systems: A Review. *International Journal of Computational Engineering Research*, 3(5):47–52, 2013.

[39] F. Ricci, L. Rokach, and B. Shapira. *Recommender Systems Handbook*. Springer, Boston, MA, USA, 2015.

[40] T. Robal, H. Haav, and A. Kalja. 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*, volume 4802 of *LNCS*, pages 170–179, Berlin, 2007. Springer.

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3): 211–252, dec 2015. ISSN 0920-5691. doi: 10.1007/s11263-015-0816-y.

[42] R. Saga and Y. Duan. Apparel goods recommender system based on image shape features extracted by a CNN. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2365–2369. IEEE,

2018. doi: 10.1109/SMC.2018.00406.

[43] G. Salton and C. Buckley. Term-Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[44] C. Shani, J. Zarecki, and D. Shahaf. The lean data scientist: Recent advances toward overcoming the data bottleneck. *Commun. ACM*, 66(2):92–102, jan 2023. ISSN 0001-0782. doi: 10.1145/3551635.

[45] P. Sheridan, M. Onsjö, C. Becerra, S. Jimenez, and G. Dueñas. An ontology-based recommender system with an application to the Star Trek television franchise. *Future Internet*, 11(9):182, 2019.

[46] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[47] R. Sun, X. Cao, Y. Zhao, J. Wan, K. Zhou, F. Zhang, Z. Wang, and K. Zheng. Multi-modal knowledge graphs for recommender systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 1405–1414, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3411947.

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. IEEE, 2015. doi: 10.1109/CVPR.2015.7298594.

[49] H. Tuinhof, C. Pirker, and M. Haltmeier. Image-based fashion product recommendation with deep learning. In G. Nicosia, P. Pardalos, G. Giuffrida, R. Umeton, and V. Sciacca, editors, *Machine Learning, Optimization, and Data Science*, pages 472–481, Cham, 2019. Springer International Publishing.

[50] V. Turner, J. F. Gantz, D. Reinsel, and S. Minton. The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. International Data Corporation, White Paper, IDC_1672, 2014.

[51] L. H. van Huijsduijnen, T. Hoogmoed, G. Keulers, E. Langendoen, S. Langendoen, T. Vos, F. Hogenboom, F. Frasincar, and T. Robal. Bing-CSF-IDF+: A semantics-driven recommender system for news. In J. Darmont, B. Novikov, and R. Wrembel, editors, *24th European Conference on Advances in Databases and Information Systems (ADBIS 2020), New Trends in Databases and Information Systems*, volume 1259 of *CCIS*, pages 143–153. Springer, 2020. doi: 10.1007/978-3-030-54623-6_13.

[52] C. Van Rijsbergen, S. Robertson, and M. Porter. *New Models in Probabilistic Information Retrieval.* British Library research & development report. Computer Laboratory, University of Cambridge, Cambridge, England, 1980.

[53] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and Tell: A Neural Image Caption Generator. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015)*. IEEE, 2015.

[54] P. D. Wasserman and T. Schwartz. Neural networks. II. What are they and why is everybody so interested in them now? *IEEE Expert*, 3(1):10–15, Spring 1988. ISSN 0885-9000. doi: 10.1109/64.2091.

[55] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A Survey of Transfer Learning. *Journal of Big Data*, 3(1):9, 2016. ISSN 2196-1115.

[56] B. Yang, J. Wu, and G. Hattori. Facial expression recognition with the advent of face masks. In *19th International Conference on Mobile and Ubiquitous Multimedia*, MUM '20, page 335–337, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450388702. doi: 10.1145/3428361.3432075.

[57] Y. C. Yoon and J. W. Lee. Movie recommendation using metadata based word2vec algorithm. In *2018 International Conference on Platform Technology and Service (PlatCon)*, pages 1–6. IEEE, 2018.

[58] L. Yu, F. Han, S. Huang, and Y. Luo. A content-based goods image recommendation system. *Multimedia Tools and Applications*, 77:4155–4169, 2018.

[59] G.-Q. Zhang, G.-Q. Zhang, Q.-F. Yang, S.-Q. Cheng, and T. Zhou. Evolution of the Internet and its Cores. *New Journal of Physics*, 10(12):123027, 2008.

[60] Q. Zhang, J. Wang, H. Huang, X. Huang, and Y. Gong. Hashtag recommendation for multimodal microblog using co-attention network. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3420–3426, 2017. doi: 10.24963/ijcai.2017/478.

[61] X. Zhou, D. Qin, L. Chen, and Y. Zhang. Real-time context-aware social media recommendation. *The VLDB Journal*, 28(2):197–219, 2019. ISSN 1066-8888.

[62] X. Zhou, D. Qin, X. Lu, L. Chen, and Y. Zhang. Online social media recommendation over streams. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 938–949, Piscataway, New Jersey, 2019. IEEE.