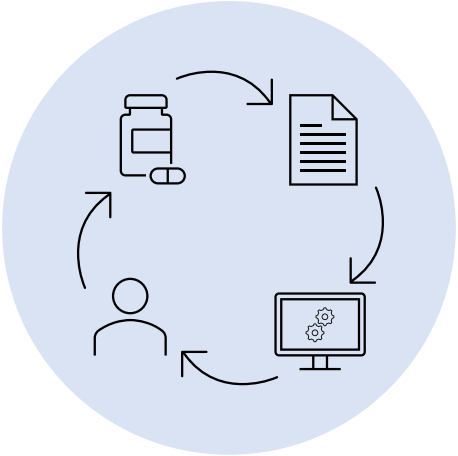
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia de Eletrónica e Telecomunicações e de Computadores



Med2Care - Recomendação de novas terapêuticas

Renato de Sousa Marcelo

Licenciado

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Professora Doutora Matilde Pós-de-Mina Pato
Professor Doutor Nuno Miguel Soares Datia

Júri:

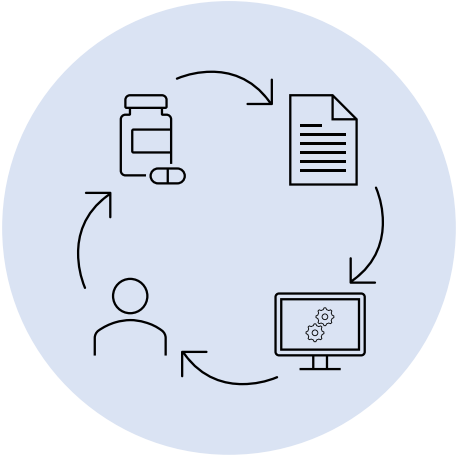
Presidente: Professor Doutor Carlos Jorge de Sousa Gonçalves

Vogais: Professora Doutora Márcia Afonso Barros
Professora Doutora Matilde Pós-De-Mina Pato

Outubro, 2023

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Departamento de Engenharia de Eletrónica e Telecomunicações e de Computadores



Med2Care - Recomendação de novas terapêuticas

Renato de Sousa Marcelo

Licenciado

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Professora Doutora Matilde Pós-de-Mina Pato
Professor Doutor Nuno Miguel Soares Datia

Júri:

Presidente: Professor Doutor Carlos Jorge de Sousa Gonçalves

Vogais: Professora Doutora Márcia Afonso Barros
Professora Doutora Matilde Pós-De-Mina Pato

Outubro, 2023

Aos meus orientadores agradecer pela incansável ajuda e apoio que me deram durante o desenvolvimento da tese (e também ao longo de todo o meu percurso académico até agora) e o apoio que me deram nos momentos menos bons onde a motivação e o empenho possa ter faltado. Agradecer ao ISEL pela experiência incrível que me proporcionou e por ter sido a minha "segunda" casa durante estes últimos cinco anos. Um agradecimento especial a alguns colegas de curso (eles sabem bem quem são) pelo companheirismo e entreajuda que tiveram comigo (e pelas muitas noites sem dormir porque não queríamos só terminar os trabalhos práticos). Para além de serem colegas de trabalho são, obviamente, amigos que levo para a vida e com bastante orgulho. Por fim, mas nunca menos imporante agradecer à minha família por me aturarem todo este tempo a falar de coisas espetaculares para mim que pareciam certamente chinês para eles, e pelo esforço (que sei que foi enorme) me proporcionarem esta experiência.

Acknowledgments

I would like to thank my advisers for their immeasurable help and support during my thesis (as well as during my academic path until now) and the incredible support and help during less happy moments when I lacked motivation and commitment. Thank ISEL for giving me this amazing experience and for being my "second" home for the past five years. A special thanks to some of my academic colleagues (they know who they are) for their companionship and help (and for the numerous nights without sleep just because we didn't want to deliver the practical projects and assignments). Besides being amazing colleagues they are, obviously, friends that I earned for life and I'm so proud of that. Last but not least, thank you to my family for putting up with me all this time talking about spectacular things that certainly seemed Chinese to them, and for the effort (which I know was huge) to give me this experience.

Abstract

Patient Information Leafletss (PIL) are responsible for containing clear and detailed information about safe and efficient medication use. With the constant increase of available medications and drug combinations, PIL information misinterpretation is facilitated. One of problem possible approaches consists of using a Knowledge Graph-based Recommender System (KGRS) in conjunction with Natural Language Processing (NLP) to clarify and simplify possible adverse drug reactions. A KGRS uses a large-scale knowledge graph to model complex relations between diseases, drugs, and other associated entities. Based on our research, this is the first attempt to explore the simplification of Patient Information Leaflets (PIL) information. With the use of these technologies, we can improve medical decision-making and recommendation processes.

Keywords: Recommender System; Patient Information Leaftlet; Active Ingredients; Natural Language Processing; Knowledge-Graph

Resumo

As bulas são responsáveis por manter informação clara e detalhada sobre a utilização segura e eficaz da medicação. Com o crescente volume de medicamentos disponibilizados e combinações dos seus princípios ativos, torna-se cada vez mais fácil interpretar incorretamente as informações presentes nas bulas. Uma das possíveis formas de abordar estes problemas passa pela utilização de *Knowledge Graph-based Recommender System (KGRS)* em conjunção com *Natural Language Processing (NLP)* para clarificar e simplificar possíveis reações adversas aos princípios ativos. Um KGRS utiliza um vasto grafo de conhecimento por forma a modelar as relações complexas entre doenças, princípios ativos e outras entidades associadas. Até ao momento esta consta ser a primeira tentativa de exploração desta abordagem de simplificação das informações presentes nas bulas. Com o uso destas tecnologias o processo de recomendação e decisão médica pode ser melhorado.

Palavras-chave: Sistemas de recomendação; Folheto informativo do medicamento; Princípios Ativos; Processamento de Linguagem Natural; *Knowledge Graph*

Contents

List of Figures

List of Tables

List of Listings

Acronyms

ADR Adverse Drug Reaction. 1, 2
AI Artificial Intelligence. 5, 26
ALS Alternating Least Squares. 6, 31, 32, 36, 37
API Application Programming Interface. 15
ATC Anatomical Therapeutic Chemical. 9, 15, 16, 17

BoW Bag of Words. 10
BPR Bayesian Personalized Ranking. 6, 31, 32, 36, 37

CAMet Classification Accuracy Metrics. 32
CB Content Based. 6, 7, 31, 32, 37
CDER Center for Drug Evaluation and Research. 5
CF Collaborative Filtering. 6, 7, 31, 32
ChEBI Chemical Entities of Biological Interest. 2, 28, 30, 37
CNN Convolutional Neural Network. 12
CSV Comma Separated Values. 28, 29

DDI Drug-to-Drug Interaction. 1, 2, 11, 12, 25, 29, 43
DO Disease Ontology. 2, 28

EHR Electronic Health Record. 11, 12
EMA European Medicines Agency. 5
EMC Electronic Medicine Compendium. 2, 10, 15, 16, 22, 24, 26, 28, 29, 35, 37, 43
EMR Electronic Medical Records. 10

HIV Human Immunodeficiency Virus. 15, 16, 24, 26, 28
HTML HyperText Markup Language. 14, 15, 16, 18, 24

ICD-10 International Classification of Diseases 10th Revision. 9
InChI International Chemical Identifier. 30

JSON JavaScript Object Notation. 20, 22, 26, 28, 29

KG Knowledge Graph. 29, 30
KGRS Knowledge Graph-based Recommender System. ix, xi, 2, 7

LSVC Linear Support Vector Classifier. 10

MF Matrix Factorisation. 6
MIMIC Medical Information Mart for Intensive Care. 11
ML Machine Learning. 1, 5, 26
MPPN Message Passing Neural Network. 11
MRR Mean Reciprocal Rank. 32, 33, 37

NA Not Available. 25
nDCG Normalised Discount Cumulative Gain. 32, 33, 37
NER Named Entity Recognition. 6, 26, 27, 28
NLP Natural Language Processing. ix, xi, 1, 2, 13, 26, 29, 30

OCR Optical Character Recognition. 14

PDF Portable Document Format. 14, 15
PI Package Insert. 5
PIL Patient Information Leaflets. ix, 1, 2, 5, 13, 14, 26, 28, 43
PIL4KGRS Patient Information Leaflets for Knowledge Graph-based Recommender System. 2

RAMet Rank Accuracy Metrics. 32
RE Relation Extraction. 6
RNN Recurrent Neural Network. 12

RS Recommender System. 2, 6, 7, 25, 28, 29, 30, 43

SKOS Simple Knowledge Organization System. 9
SMILES Simplified Molecular-Input Line-Entry System. 30
SmPC Summary of Product Characteristics. 5

TF-IDF Term Frequency–Inverse Document Frequency. 10

UNII Unique Ingredient Identifier. 9
URL Unique Resource Location. 17, 18, 19
USFDA United States Food and Drug Administration. 5

WHO World Health Organization. 1
WHOCC World Health Organization Collaborative Center. 16

XML Extended Markup Language. 15, 24

1

Introduction

Patient Information Leafletss (PILs) are a valuable information source for Machine Learning (ML) applications, with more emphasis on those who deal with Natural Language Processing (Natural Language Processing (NLP)) and text mining. They contain practical information such as:

- Important drug information, like brand name, active ingredients, and intended use;
- Information about interactions with other medications, food, or substances.

In summary, PILs' principal objective is to provide helpful information to increase patients' awareness of proper use and possible side effects.

Even if a PIL is well-written, there are some ways where its information can still be misunderstood, leading to (1) medication misuse; (2) overdose; (3) Drug-to-Drug Interaction (DDI); (4) Adverse Drug Reaction (ADR); (5) disregard of side effects; and (6) self-diagnosis.

According to World Health Organization (WHO), medication errors are estimated to affect 1/30 of worldwide patients, leading to disability, harm, or even death. Furthermore, the leading reasons for these occurrences are poor literacy, language barriers, and a lack of understanding of medical terms and instructions. Is also concluded that the most common adverse events that may result in avoidable patient harm are medication errors [28, 65].

We trust that NLP can and may support reducing misused PIL instances by enriching and improving contained information, making it more understandable for patients. Enumerating some examples:

1. NLP algorithms specialized in improving readability can process and analyze PIL-used language to ensure that it is simple and understandable for patients;
2. NLP algorithms for language barrier detection can provide automatic translation services, making information accessible to non-native speakers;
3. NLP algorithms for patient data analysis can provide personalized information about their condition, including possible DDIs and side effects;
4. NLP algorithms can analyze PILs to allow error detection and correction, like incorrect dosage or contraindications.

This thesis presents a solution focused on example (3) of the previous list by creating a Knowledge Graph-based Recommender System (KGRS). This Recommender System (RS) can analyze PILs and patient data to monitor and identify possible DDIs, helping to prevent medication misuse. To our knowledge, this is the first study attempting to automate leaflet simplification. In addition, there is no publicly available dataset containing implicit references to the active principles of possible therapeutic indications. We aim to create a ready-to-use dataset using NLP techniques. Our operational goal is to use large document *corpora* like the one accessible in Electronic Medicine Compendium (EMC) to create a final personalized information dataset. In particular, we focus on the lexical simplification of ADR described in these documents.

Our approach generates a typical <user, item, rating> dataset, where the active ingredients represent the users, given by Chemical Entities of Biological Interest (ChEBI), and therapeutic indications, as well as contraindications, correspond to the item, identified by Disease Ontology (DO).

Despite our dataset structure being the same as in [7, 8, 51], the meaning attributed to the user, item, and rating is significantly different. Ortega et al [51] built a dataset to recommend topics from an article and articles from a given topic. In contrast, Barros et al [7, 8] recommend scientific items to people based on their peers. We also provide an open dataset called PIL4KGRS (Patient Information Leaflets for Knowledge Graph-based Recommender System) created using Python.

Sample datasets, as well as all developed code, are available publicly at `https://github.com/matpato/PIL-TMI`. Also, this thesis produced an article named "A

PIL-based knowledge graph recommendation system" that was submitted to BMC Medical Informatics and Decision Making journal, with the submission code 'c3dfae66-a395-4982-8ef0-ab76eb3a8384', and is sent with this thesis report.

The remaining document is structured as follows: (1) chapter 2 discusses the performed research work; (2) chapter 3 discusses the existing background work; (3) chapter 4 discusses the proposed method developed to attain the objective; (5) chapter 5 presents and discusses the achieved results and points out future work directions; Appendix A presents all obtained results for the generated datasets.

2

Background

The importance of having medical information freely available is undeniable for several reasons. For example, one of the reasons is to combat systemic mistrust and false information [19], enhance research through the use of Artificial Intelligence (AI) and Machine Learning (ML) [37] to resume and suggest the most relevant information pieces [44], towards trustworthy medical intelligence [29]. Moreover, the European Union, in its research and innovation funding program Horizon Europe, has a mandatory open access policy for publications and open science principles.

In the case of medication information provided to the general public, there are two international systems that provide this information. The first system is the Package Insert (PI), which follows the guidelines of the United States Food and Drug Administration (USFDA) and offers information for both patients and healthcare professionals [23]. The second system comprises the PIL, released by the European Medicines Agency (EMA) for patient use, and the Summary of Product Characteristics (SmPC), intended for healthcare professionals [15, 16]. Although it is in the public domain, digital media makes it challenging to use for research. As an example, access is limited to search fields. In 2022, the FDA's Center for Drug Evaluation and Research (CDER) approved 37 novel medicines [18], and EMA recommended for authorization 89 new medicines [14]. This increase in total medicine number and medicine complexity generates large datasets that are challenging for researchers to explore in detail, that said, information analysis is a time- and effort-consuming procedure.

To address these challenges, there is a need for medical information automated extraction from unstructured texts. The primary technique for obtaining data from documents is Named Entity Recognition (NER), followed by Relation Extraction (RE). In NER, the beginning and last characters that refer to an entity in the text are determined by their offset. Various studies on biomedical NER have been done about different entities [68], such as genes, chemicals, and diseases [5, 39, 42]. Finding relationships between entities, stated in a particular document is RE's goal.

Current RS already showed the importance in the biomedical field on different challenges, such as models designed to identify unnecessary hemoglobin (Hgb) tests for hospitalized patients [32], recommending medicine to combat SARS-CoV-2 during the COVID-19 pandemic [24], suggesting entities of potential interest to specific researchers [9].

RS can be a viable solution by suggesting medicines based on user diseases, medical history, and other relevant information. One approach is sentiment analysis-based medicine recommendation systems that analyze patients' medication reviews and forecast the underlying sentiment expressed. Here, these predictions suggest the best medicine for a particular condition [22, 30]. However, recommending medicines has yet to be widely studied [50, 62, 64]. One of the challenges in incorporating RSs into medicine is the need for more available datasets to evaluate them. For example, determining the appropriate medication for a specific disease can be challenging. However, new alternatives have emerged with the appearance of implicit feedback datasets [7, 31, 49]. Unlike well-known datasets such as Movielens [27] or Netflix [4], which contain explicit feedback in the form of user ratings, these datasets do not directly contain users' explicit interests. Depending on dataset type and application requirements, different algorithms are available for RSs. Alternating Least Squares (ALS) is a robust Matrix Factorisation (MF) algorithm focused in explicit and implicit feedback RSs [53, 57, 63]. On the other hand, Bayesian Personalized Ranking (BPR) [9, 57] is only focused on implicit feedback datasets.

Popular algorithms for RSs include Collaborative Filtering (CF), Content Based (CB) Filtering, and hybrid models that combine both previous methods [59]. The Collaborative Filtering (CF) techniques utilize preference information obtained from a set of users to predict a user's interests automatically. Simultaneously, it utilizes similarities between users and items to provide recommendations. In contrast, Content Based (CB) recommends similar items to what the user likes based on their past actions or explicit feedback, utilizing item features. Recommendations by CB follow assigning keywords and attributes to objects in a database and matching them to a user profile. Hybrid models combine both previous strategies to utilize their benefits. Hybrid RS can also

address challenges inherent to the previous RS approach, e.g. the cold start problem, present for new items in CF and new users in CF and CB. The models are capable of using CF and CB techniques to generate recommendations with enhanced precision [59].

Knowledge Graph-based Recommender System (KGRS) have been developed to address information overload problems and improve user experience in various online applications [17, 26]. These systems use knowledge graphs for more accurate and understandable recommendations. Recommendation systems can overcome challenges such as data sparsity and the cold start problem that traditional models suffer from [26].

3

State of the Art

This chapter describes other works and studies done to achieve the same or similar objectives.

Charalampos Doulaverakis et al. presented a semantic-enabled service, GalenOWL [20], capable of offering real-time drug-drug and drug-disease interaction discovery. They base their work on top of already existing models like International Classification of Diseases 10th Revision (ICD-10), Unique Ingredient Identifier (UNII), and Anatomical Therapeutic Chemical (ATC). Doulaverakis et al. present in their results that the model responds efficiently to the set of test queries, having the drawback of needing an average of 148 seconds to perform the model initialization phase. Another disadvantage lies in the resources necessary to maintain the model, around 649 MB of system memory, which makes the model somewhat impractical as the volume of rules increases.

Later, Doulaverakis et al. complemented their work on GalenOWL, adding standardized medical terms and rich knowledge organized in Simple Knowledge Organization System (SKOS) vocabulary using ontology with rule-based and medical reasoning approaches, naming the new framework Panacea [52]. They concluded that Panacea is a promising alternative but still needs some improvement.

SemMed [61] consists in an ontology-based recommendation engine approach. Their work focused on "Diseases", "Medicines" and "Allergies" ontology classes to develop the engine recommendation rules. Their initial system reveals how a medical decision support system is really offered because not only provides recommendations but also

eases healthcare professional tasks.

Focusing now on the second approach, Leilei Sun et al. [12] analyzed EMC records to obtain information on typical treatment regimens and evaluate (quantitatively) their effectiveness for particular patient cohorts. They also measure the similarity between Electronic Medical Records (EMR) records, clustering similar treatment regimens and extracting useful information. This information is used to estimate a treatment outcome for a patient cohort. Sun et al. conclude that with this approach the effective rate of the patient increases as well as the cure rate.

Another study by Nidhi Kushwaha et al [43] described a drug recommender system based on semantic web technology and data mining algorithms. They combined these two methodologies to extract the semantic information and apply data mining algorithms to that data. Their approach to data mining was to individualize treatment from the patient attributes, making a system not recommend drugs that the patient already took or that may interact with previous medications.

Another approach proposes a hybrid drug recommender framework, where practitioners enter patient information as a new case and the system processes the new data to extract patient features. After obtaining these features, the system creates a diagnosis and matches it to a given disease category to determine the symptom-drug classifier to use. After defining the adequate drug set to recommend, a ranking order is performed and presented as a recommendation list [33].

On a different approach, Satvik Garg [56] intends to present a drug recommender system created on patient reviews and predicted sentiment analysis. Makes use of well-known vectorization techniques, such as Bag of Words (BoW), Term Frequency–Inverse Document Frequency (TF-IDF), Word2Vec, and Manual Feature Analysis to help recommend the top drugs for a given disease. After testing various classification algorithms, he concluded that the best combination for the vectorization process and classification algorithm was Linear Support Vector Classifier (LSVC) using TF-IDF with 93% accuracy; Deloar Hossain et al [55] used the same approach to implement a recommender system framework, adding a rating generation procedure based on sentiment analysis from drug reviews. They also conclude that LSVC achieves the best results and offers the best trade-off between accuracy and model efficiency/scalability.

The previous approaches focused more on providing drug recommendations to patients. The following studies focus on developing a medication-oriented recommendation to aid medical professionals in prescribing the correct medications for each patient diagnosis.

Some of the following articles mention the use of Medical Information Mart for Intensive Care (MIMIC) [47] dataset (also known as MIMIC-III). This corresponds to a large, single-center database containing information related to admitted patients in critical care units. Contains vast amounts of clinical information such as vital signs, medications, observations, diagnostic codes, etc.

SafeDrug [60] is a drug-drug interaction-oriented recommender model that also leverages drug molecule structure. This model uses an Message Passing Neural Network (MPPN) module and a local bipartite learning module to encode molecule connectivity and functionality. Its DDI functions allow for a more efficient drug combination recommendation. All these features make SafeDrug outperform previous approaches, requiring fewer parameters over other deep learning-based approaches, which leads to faster training and inference. These inference results don't take into account possible chemically similar active principles to enrich their recommendation model.

COGNet [67] introduces a copy-or-predict mechanism to generate an adequate medicine set for patient diagnosis, preventing harmful drug-drug interactions. This mechanism retrieves historical diagnoses and medication recommendations, extracting relations with current diagnoses. Furthermore, the model then decides whether to copy a previous medicine or to predict a new one, simulating the doctor's decision process. Then, using the public MIMIC [47] dataset, they evaluate the proposed model, concluding that it outperforms state-of-the-art approaches, like SafeDrug and MICRON [46].

4SDrug [1] is a symptom-based set-to-set safe and small drug recommender framework. This framework supports drug and symptom set comparison, using defined set-oriented representation and similarity measure. Some emphasis is made on set techniques for increased quality and safety. Devise importance-based set aggregation to enhance representation accuracy on symptom sets and intersection-based set augmentation to ensure smaller safe drug sets. Furthermore, extensive experiments using MIMIC-III public dataset and enterprise NELL dataset claim 4SDrug to outperform all baselines in most effectiveness measures, yielding small recommended drug sets with significant DDI rate reduction from ground-truth data.

GAMENet [21] integrates DDI knowledge graphs within graph convolutional networks and uses patient records as model queries, providing safe and personalized medicine combination recommendations. Using DDI data, Junyuan et al. demonstrate its effectiveness, outperforming all baselines in effectiveness measures. On the other hand, achieves low DDI reduction on existing Electronic Health Record (EHR) compared to the previous two approaches.

MetaPred [45] consists of a meta-learning model for clinical risk prediction from longitudinal patient EHR. Due to limited data samples, a meta-learner is trained from related risk prediction tasks to be directly used in risk prediction tasks, using the aforementioned data samples to improve models' performance. The results concluded that in conjunction with Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) as base predictors, MetaPred achieves much better performance predicting target risk, using low resources compared with the predictor trained on the limited sample information.

Decagon [13] models polypharmacy side effects, constructing a multi-modal graph of protein-protein, drug-protein, and drug-drug interactions. One of Decagon's main advantages is the specialization of large multi-modal graph edge types. Decagon is developed as a new graph CNN for multi-relational link prediction in the aforementioned multi-modal networks. Unlike approaches limited to predicting simple drug-drug interactions, claims that can predict exact side effects through drug combination manifest. Its results claim that it can outperform baselines by up to 69%. Another result shows that it can automatically learn DDI representations from patients' co-occurrence. Furthermore, Decagon models well DDI based on a heavy molecular basis, while still achieving good results in non-molecular side effects because of sharing of model parameters across edges.

4

Proposed Method

This chapter discusses the proposed method for building a medicine recommender system dataset.

We present the general conceptual architecture in Figure 4.1.

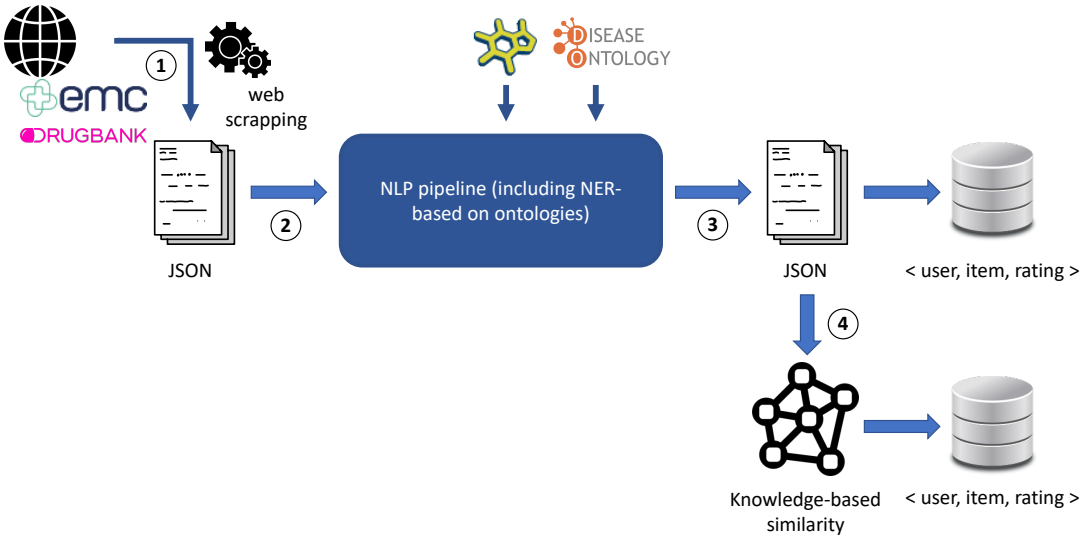


Figure 4.1: The workflow for constructing PILRS to create a < user, item, rating > dataset based on Patient Information Leaflets. ① web scrapping, ② NLP pipeline, ③ entity extraction and ④ knowledge graph construction.

4.1 Web-scraping and information extraction

In step ① it is necessary to acquire data, more precisely, drug information. As mentioned before, most of the medicine data used in recommender systems are associated with sentiment analysis, but in our particular problem, this data has no use. With that in mind, we needed to find a source of drug information adjusted for health professional's use.

Before entering into web scrapping module details, it is important to mention all the thoughts that went into deciding to develop this approach. The first data source option was **Infarmed** - Autoridade Nacional do Medicamento e Produtos de Saúde[1]. This organization is responsible for regulating drugs and healthcare products in Portugal. They provide a search engine, open to the public, where a user can search medicines by active ingredient, medicine name, or others (not so much related to our problem domain). This search engine then returns a set of matches found which the user can choose from. The downside of this data source is that the retrievable information comes in a Portable Document Format (PDF) format, which is somewhat complicated to work with as it was designed to be easy to understand by humans, and not machines. Adding to that complexity is the fact that those files don't have a defined structure for representing a PIL which is impractical for our study case [2, 38]. A context-aware text extraction over the PDFs might help to solve that [54] but it diverges from the study case scope. Some of the common problems with PDF information extraction are:

1. Complex formatting as often PDF files contain complex layouts, such as columns, headers, footers, or more visual elements like tables or even images, all of which make information extraction difficult to do accurately;

2. Text extraction issues, where different text encoding and font sizes contribute to improper or garbled resulting text information;

3. Image-based PDFs where some files are based on scanned documents or pictures taken from documents. This makes the contained text essentially an image. In those cases, the approach to extracting that text requires the use of Optical Character Recognition (OCR) tools, which may introduce errors or inaccuracies in the extraction process [38];

4. Lack of semantic structure, as unlike other document types (e.g. HyperText

[1] Available at https://www.infarmed.pt/

Markup Language (HTML), Extended Markup Language (XML)), PDF files often don't have a semantic structure and useful metadata that enrich the extraction process. A logical structure may not be explicitly defined, making element selection and retrieval very uneven and complicated;

With all this, **Infarmed** got discarded as a possible data source option.

Then, another option was to use **DrugBank** drug database as a data source. This consists of XML file that contains a set of drugs divided into two categories: small molecule drugs and biotech drugs. What impedes us from using this as a possible data source, is that access to this database is locked behind an account that requires additional approval from the University of Alberta (owner and maintainer of **DrugBank**), so it became a secondary data source, described in more detail in Section 4.2.

Another considered option was using **MedlinePlus** website. This website contains information about over 1100 active ingredients, a considerable amount from where to choose. The downside of using this website is that a web scraping module needs to be implemented to extract said information and the information present in **MedlinePlus** is not the most updated (the most recent updates are from August 2022).

At last, we looked into the EMC website. This online compendium contains related information to all medicines available in Europe, a good starting point for obtaining the needed information, so we started here. This information is only available for consultation via their website, without an accessible Application Programming Interface (API). So, we need to develop a new approach, and after some discussion, we agreed on a web scraping approach. This approach uses web scraping tools to retrieve the information from web pages, more precisely in the HTML elements. While the approach is the same as for MedlinePlus, the information structure present on EMC website is much better organized, when compared to MedlinePlus.

As a proof-of-concept, we initially focus our data extraction on one particular condition, Human Immunodeficiency Virus (HIV), to reduce the set of active principles, which is massive for all known diseases.

Each active ingredient has a unique code that can identify it to every health organization, named ATC. This code is assigned to the active ingredient according to the organ or system it works and how it works, which is satisfactory for creating a search query for drugs with that specific ATC code.

4.1.1 Obtaining active ingredients names

With that, we need to define which active ingredients are appropriate for treating the HIV disease. After various attempts to define a search query for retrieving all the appropriate active principles (inside EMC) and no visible results, we needed to find another approach to fix this problem.

After some research, we ended up finding a website that maintained a human disease database, named MalaCards[2]. This database contains a disease search engine and with the results, we could extract the associated active ingredient names (e.g. abacavir) into a text file named after the disease (e.g. hiv.txt). This solved our problem relatively well. It is essential to point out that in this step the process is manual, and future iterations could be automated, but it was not the main focus of the current dissertation.

4.1.2 Transforming the names of the active ingredients into the respective ATC codes

At this step, we need to transform the active ingredient names into the respective ATC codes. For that, we can use the World Health Organization Collaborative Center (WHOCC)[3] to query for the ingredient name and retrieve the respective ATC code. If done manually, this process can be very time-consuming and tedious, even with a reasonably small name set, so a scraping approach was implemented. We need to keep in mind our choices for scrapping this website because it is a single-page application, narrowing down the possible libraries to use. We ended up using the **Selenium**[4] python library for being simple and robust to deal with single-page applications because instead of scraping the HTML responses directly, it behaves like a browser (headless browser). A headless browser consists of a typical browser but with its graphical interface removed.

Now explain the complete implemented process for this step. We first load all the files created in 4.1.1, getting all the active ingredient names. Next, we pass this active ingredient name to the headless browser to perform the search. After receiving the results, we need to find the HTML element that contains the ATC code and extract its value. The process repeats for all the remaining names. Finally, we save the resulting ATC code into the respective files organized the same way as mentioned in **??**.

[2]Available at https://www.malacards.org/
[3]Available at https://www.whocc.no/atc_ddd_index/
[4]Available at:https://www.selenium.dev/

4.1.3 Retrieving drug information from EMC

After finding the set of ATC codes, a search query must be constructed capable of returning all the associated medicines. After some fuzzing and exploration, some query parameters and most importantly, the search endpoint, were found to perform the correct search. This corresponds to the following:

- search endpoint:
  - /emc/search
- query parameters:
  - **q**, allows the search for terms, in this case, a particular ATC code.
  - **filters**, allows the definition of search filters to direct the results (e.g. 'q', 'filters', 'fullText').

Before it is possible to obtain the medicine data, we need to perform a two-step approach. The first step consists of constructing a set of Unique Resource Location (URL)s corresponding to all the search results of medicines with the previously identified active ingredients. In the second step, we obtain all the necessary data.

Focusing on the first step, we use the search endpoint and the associated parameters to better concise our search, ending up with a search URL template like the following: `https://www.medicines.org.uk/emc/search?q=<atc-code>&offset=<result-offset>&limit=<max-results>&fullText=<true|false>`

Explaining a bit more what each of these parameters represents:

- '**q**' represents a text query (in this case, the ATC code);
- '**offset**' represents defining the result offset to support multi-page searches;
- '**limit**' represents the max results to obtain at once;
- '**fullText**' specifies if the text query has to be an exact match or can be a partial one.

With this URL set constructed, we perform a web scraping approach to extract the medicine URLs from the results.

For this, we chose the **Scrapy**[5] Python library. What is interesting in this library, and one of the reasons for its use is that the scraping process is encapsulated on **Spider/Crawler** components, promoting a more organized and concise approach. **Crawlers** are appropriate in situations where all the work needed is to follow links and retrieve the data from these links. On the contrary, if you need to have more complex work, **Spiders** offer a more flexible approach to dealing with navigation and extraction. In this step, we used a Spider to take advantage of its flexibility in multi-page result processing.

Focusing now on the second step, using the previously obtained URLs, we need to perform web scraping techniques to retrieve the medicine information. This data consists of the following:

- **Identifier**, for uniquely identifying a particular medicine;
- **Name**;
- **Composition**;
- **Clinical Particulars** that contain the medicine clinical particularities;
  - **Therapeutic Information** indicating the use cases for the medicine;
  - **Contraindications** containing all the medicine-known contraindications;
    - **Disease** containing all diseases that make this medicine use incompatible;
    - **Pregnancy**, for complications associated with medicine use;
    - **Machine Operation**, for complications associated with machine operation (e.g. driving);
    - **Excipients**, for a list of present excipients that can cause an adverse reaction;
  - **Revision Date** indicates when the last information update occurred (helpful for maintaining control of possible duplication of information and version control.

Observing the common HTML page structure, something pops out. An HTML element with a well-defined identifier appears right before every section. So, it can be used for more dynamic access to the elements and to extract their contents. On the other hand, the HTML page structure does not implement a parent-children approach

[5] Available at https://scrapy.org/

for representing each section, making access to the information elements messier. To make this process more dynamic and clean, we opted for utilizing the **BeautifulSoup**[6] library, instead of the **Scrapy** embedded one. **BeautifulSoup** is a popular Python library used to perform web scraping. Its main advantage over **Scrapy** comes from its flexibility (and simplicity) in terms of navigating through the parsed content. With that in mind, we implemented a similar approach as the previous step, defining a **Spider** component responsible for scraping the medicineURL response and obtaining the referred data using the **BeautifulSoup** library.

This component follows a simple structure, containing a central function, that receives each response, presented in Listing 4.1, and a separate function to deal with each data portion. To reduce the code repetition, we implemented an auxiliary function responsible for retrieving text from a given element, present in Listing 4.2.

Listing 4.1: Web scrap parse function

```
def parse(self, response: HtmlResponse, **kwargs):
    from hashlib import sha1

    parser = BeautifulSoup(response.body, features='lxml')

    medicine_id: str = str(re.findall('[0-9]+', response.url)[0])
    medicine_name = parse_medicine_name(parser)
    composition = parse_composition(parser)
    clinical_particulars = parse_clinical_particulars(response)
    revision_date = parse_revision_date(parser)

    hashed_id = sha1(f'{medicine_id}{revision_date}'.encode()).
↪ hexdigest()

    medicine: Medicine = {
        medicine_id=hashed_id,
        emc_id=medicine_id,
        metadata={
            name=medicine_name,
            composition=composition,
            clinical_particulars=clinical_particulars,
            revision_date=revision_date
        }
    }
    yield medicine
```

Listing 4.2: Extract EMC page content function

[6] Available at https://beautiful-soup-4.readthedocs.io/en/latest/

```python
def extract(anchor_id: str, parser: BeautifulSoup, return_element: bool = False) -> str | Tag:
    match = parser.find('summary', attrs={'id': anchor_id})
    content_div = match.find_next('div', attrs={'class': 'sectionWrapper'})
    if return_element:
        return content_div
    text: str = ''
    for s in content_div.strings:
        if s.isprintable():
            text = f'{text}\n{s}'
    text = text.strip()
    return text
```

With all obtained data, we create a Medicine object, presented in JavaScript Object Notation (JSON) format in Listing 4.3.

Listing 4.3: JSON schema for medicine information

```
{
    "medicine_id": "<medicine_id>",
    "emc_id": "<emc medicine_id>",
    "metadata": {
        "name": "<medicine name>",
        "composition": "<active principle with amount>",
        "clinical_particulars": {
            "therapeutic_indications": "<therapeutic indications for usage>",
            "contraindications": {
                "disease": "<interactions to preexistant diseases>",
                "pregnancy": "Applicable | Not Applicable",
                "machine_ops": "Applicable | Not Applicable",
                "excipients": "<semicollon separated excipients present in medicine>",
                "incompatibilities": "<semicolon separated medicine names that are incompatible with this medicine> | Not Applicable"
            },
        "revision_date": "<revision date with format yyyy-mm-dd>"
    }
}
```

4.1.4 Cleaning extracted documents

In this step, we proceed to clean and normalize the medication information.

Here, we apply two cleaning/normalization approaches, removing non-printable characters and normalizing the date format. These approaches are isolated in two distinct functions, normalize_general and nomalize_date, present in Listing 4.4 and Listing 4.5.

Listing 4.4: Normalize medicine text fields function

```
def normalize_general(data: dict) -> dict:
    normalized_data: dict = copy.deepcopy(data)

    name: str = get_member_recursive(normalized_data, 'name')
    composition: str = get_member_recursive(normalized_data, 'composition')
    therapeutic_indications: str = get_member_recursive(normalized_data, '
    ↪ therapeutic_indications')
    disease: str = get_member_recursive(normalized_data, 'disease')

    name = all_normalizations(name)
    composition = all_normalizations(composition)
    therapeutic_indications = all_normalizations(therapeutic_indications)
    disease = all_normalizations(disease)

    set_member_recursive(normalized_data, 'name', name)
    set_member_recursive(normalized_data, 'composition', composition)
    set_member_recursive(normalized_data, 'therapeutic_indications',
    ↪ therapeutic_indications)
    set_member_recursive(normalized_data, 'disease', disease)

    return normalized_data
```

Listing 4.5: Normalize medicine date function

```
def normalize_date(data: dict) -> dict:
    revision_date: str = get_revision_date(data)
    try:
        revision_date = all_normalizations(revision_date)
        revision_date = ' '.join(revision_date.split(' ')[:3])
        revision_date = str(parser.parse(revision_date).date())
        return set_revision_date(data, revision_date)
    except Exception as date_error:
        print('Cannot auto parse date field')
        print('Trying particular parse')
        try:
            revision_date = ' '.join(revision_date.split(' ')[:2])
            revision_date = str(parser.parse(revision_date).date())
            return set_revision_date(data, revision_date)
        except Exception as date_error_part:
            print('ERROR: There is a problem with revision date field')
```

```
17                print(date_error)
18                print(date_error_part)
```

The first method applies the previously mentioned approach to all the medication text fields. The second method normalizes all known date formats into a uniform one (<yyyy-mm-dd>), using the embedded Python date parser. Here are some of the observed date formats that caused date comparison problems.

Table 4.1: Date inconsistent formats

Problematic Format	Uniform Format
16/11/2022	2022-11-16
09 May 2022	2022-05-09
June 2020	2020-06-01
13th May 2020	2020-05-13
08/2022	2022-08-01

This cleaning step allows us to improve text field content for the next module and normalize the date format for faster and correct duplicate detection.

4.1.5 Removing duplicate document occurrences

In this final step, we process the normalized medicine JSON files to detect and remove duplicate ones. We consider a duplicated medicine the one that has the same revision date and the medicine identifier as a previously processed one.

So, to discard any of the duplicated medicines, first, we need to compare the newly extracted medications with the current set. So, we implemented a Python function, verify_duplicate_files, that obtains the EMC ids that coincide in both groups and applies the implemented check_overlapped method to that subgroup to perform date comparison. This last function is only applied if the EMC ids are the same, as it has no use if they are different already. Listing 4.6 and Listing 4.7 show the concrete implementation for both functions.

Listing 4.6: Verify medicines duplicated ids function

```
1 def verify_duplicate_files(extracted_dir: str, processed_dir: str) -> tuple
    ↪ [int, list]:
2     extracted_ids: dict[str, str] = get_ids(extracted_dir)
3     processed_ids: dict[str, str] = get_ids(processed_dir)
```

```
4
5       df_extracted: DataFrame = DataFrame(
6           extracted_ids.items(),
7           columns=['emc_id', 'id']
8       )
9       df_processed: DataFrame = DataFrame(
10          processed_ids.items(),
11          columns=['emc_id', 'id']
12      )
13
14      overlapped: DataFrame = df_extracted[df_extracted.emc_id.isin(df_
    ↪ processed.emc_id)]
15      number_duplicates, overlapped_to_keep = check_overlapped(overlapped,
    ↪ extracted_dir, processed_dir)
16
17      new_medicines: DataFrame = df_extracted[~df_extracted.id.isin(df_
    ↪ processed.id)]
18
19      to_return: DataFrame = pd.concat([overlapped_to_keep, new_medicines])
20
21      return number_duplicates, to_return['id'].values.tolist()
```

Listing 4.7: Check revision date from overlapped medicines function

```
1   def check_overlapped(overlapped: DataFrame, extracted_dir: str, processed_
    ↪ dir: str) -> tuple[int, DataFrame]:
2       new_ids: list = []
3       count: int = 0
4       for identifier in overlapped.id.values:
5           extracted_medicine_doc: dict = read_json_file(extracted_dir,
    ↪ identifier)
6           processed_medicine_doc: dict = read_json_file(processed_dir,
    ↪ identifier)
7
8           revision_date_extracted: str = get_revision_date(extracted_medicine
    ↪ _doc)
9           revision_date_processed: str = get_revision_date(processed_medicine
    ↪ _doc)
10
11          date_extracted: date = datetime.strptime(revision_date_extracted, '
    ↪ %Y-%m-%d').date()
12          date_processed: date = datetime.strptime(revision_date_processed, '
    ↪ %Y-%m-%d').date()
13
14          if date_extracted > date_processed:
15              new_ids.append(identifier)
```

```
        elif date_extracted == date_processed:
            count += 1

    return count, DataFrame(new_ids, columns=['id'])
```

After this step, we are left with a small set of medicines indicated to treat HIV.

In a later iteration, we contacted EMC to ask for access to all information as an academic project to which they responded with a paid-for-access approach instead of embracing our project, restricting our current approach. Nevertheless, restricted access permissions hindered us from extracting information beyond medicines for HIV and Hepatitis B during data collection. Consequently, a total of 78 medicines were acquired from EMC.

4.2 Exploring DrugBank database

After requesting and obtaining access to **DrugBank** complete database, we proceed to analyze its contents and define an adequate approach to extract drug information from it. The first fact to mention is the database file size, which is almost 1.5GB, a relatively large XML file. This file is composed of a root **drugbank** element, which contains a set of **drug** elements. These latter elements are the ones we want to focus on as they represent the drugs (medicines) that are our basis of information.

Due to the large file size, our approach kept in mind that if we want to process and extract information from the database, the file has to be loaded in chunks sequentially to avoid consuming large amounts of system memory.

To achieve this processing behavior we used the 'lxml' Python library, which has iterable capabilities while processing large files (XML or HTML). This is achieved using the **iterparse** function. This function can be configured to iterate over reading events like **start** or **end** of XML elements.

In our approach, we need to iterate over both events to extract information from child elements inside each drug element.

Each **drug** element is composed of various child elements containing all kinds of information. From this element, we decided to extract the same information as the one extracted from EMC. Due to the different data organization, some processing needs to be performed to preserve the processed structure:

1. Find the drug toxicity element and associate it with disease contraindications;

2. Define pregnancy and machine_ops as Not Available (NA) in processed document;

3. no element contains information for possible excipient contraindications, so the excipients field is also defined to NA;

4. As it is possible to extract DDIs from the drug interactions element, they are extracted to the incompatibilities field.

In Listing 4.8 is presented the generic structure to extract drug information.

Listing 4.8: Drugbank drug information extraction

```
1  def extract_drugs():
2      drugbank_xml: Iterator = et.iterparse('<path_drugbank_db>', events=("
       ↪ start", "end"))
3
4      # Variable defaults declaration
5
6      # Iterate over the XML and extract information from the interesting
       ↪ tags
7      for event, elem in drugbank_xml:
8          # We reached a new medicine to be processed
9          if is_drug_start(event, elem, started):
10             # Capture drug main details (name, revision date, etc.)
11         elif is_disease_contraindications(event, element):
12             # Capture disease contraindications
13         elif is_indications(event, element):
14             # Capture therapeutic indications
15         elif is_drug_interactions(event, element):
16             # Capture drug interactions
17         elif is_drug_end(event, elem, started):
18             # Process all fields and convert to JSON file
```

Is worth mentioning that all the processed information is carried over to **elif** scope in line 17, where is checked if exists information regarding therapeutic indications, discarding the complete drug object if none is found. This is important because we cannot use a drug object that does not provide information for its uses and applications. This simple check statement reduces the original drug objects, which corresponded to approximately 13300, to a bit more than one-third, coming to 4031 distinct medicines.

It is important to mention that this structure is scalable, meaning that more document fields can be added to support other study cases, as needed. For instance, the RS could use user input and the trained model to generate personalized recommendations for medication use, dosage, and potential side effects.

```
{
  "medicine_id": "d45690e7d52c990799fef3f59fc161d3f4bc60c1",
  "platform_id": "10525",
  "metadata": {
    "name": "Abacavir 300 mg Film Coated Tablets",
    "composition": "Each film coated tablet contains 300 mg of abacavir For the full list of",
    "clinical_particulars": {
      "therapeutic_indications": "Abacavir is indicated in antiretroviral combination therapy for the treatment of Human Immunodeficiency
Virus HIV infection in adults adolescents and children see sections 4 4 and 5 1 The demonstration of the benefit of Abacavir is mainly based
on results of studies performed with a twice daily regimen in treatment na\u00efve adult patients on combination therapy see section 5 1
Before initiating treatment with abacavir screening for carriage of the HLA B 5701 allele should be performed in any HIV infected patient
irrespective of racial origin see section 4 4 Abacavir should not be used in patients known to carry the HLA B 5701 allele",
      "contraindications": {
        "disease": "Hypersensitivity to abacavir or to any of the excipients listed in section 6 1 See sections 4 4 and 4 8",
        "pregnancy": "Applicable",
        "machine_ops": "Applicable",
        "excipients": "Cellulose, microcrystalline PH102;Sodium starch glycolate (type A);Silica, colloidal anhydrous;Magnesium
stearate;Polyvinyl alcohol (E1203);Titanium dioxide (E171);Talc;Iron oxide yellow (E172);Macrogol (E1521)",
        "incompatibilities": "Not applicable"
      }
    },
    "revision_date": "2022-01-12"
  }
}
```


Figure 4.2: An example of the JSON file showing all available fields in PIL related to Abacavir, as obtained from EMC.

Figure 4.2 depicts all fields and corresponding values for Abacavir, a prescription medicine for the treatment of HIV infection in adults, children, and infants.

4.3 NLP and NER pipeline

In this section, we proceed to implement a Natural Language Processing (NLP) pipeline with Named Entity Recognition (NER), corresponding to step ② on the complete architecture. This pipeline is created to allow named entity extraction from the data previously gathered in the JSON file set.

NLP is a subfield of Artificial Intelligence (AI) focused on interpreting natural language (used between humans) and translating it to an intermediate language that can more easily be understood by ML models. This is a needed step as natural language is full of ambiguities, that make it complicated to understand the meaning of words and sentences, leading to unwanted interpretations or results [36]. So, an adequate NLP pipeline must be implemented to correctly process and break down human text, like the one present in PILs.

In Figure 4.3 it is possible to observe the complete NLP pipeline architecture that was implemented and the constituent NLP tasks.

Tokenization Given a sentence and a defined document unit tokenization, the sentence is sliced up into segments, which are called tokens. Also in this process, some characters like the punctuation ones, are filtered out. This task uses the '**nltk**' Python library with their recommended tokenizer.

Lowercase Each resulting token is converted to lowercase format.

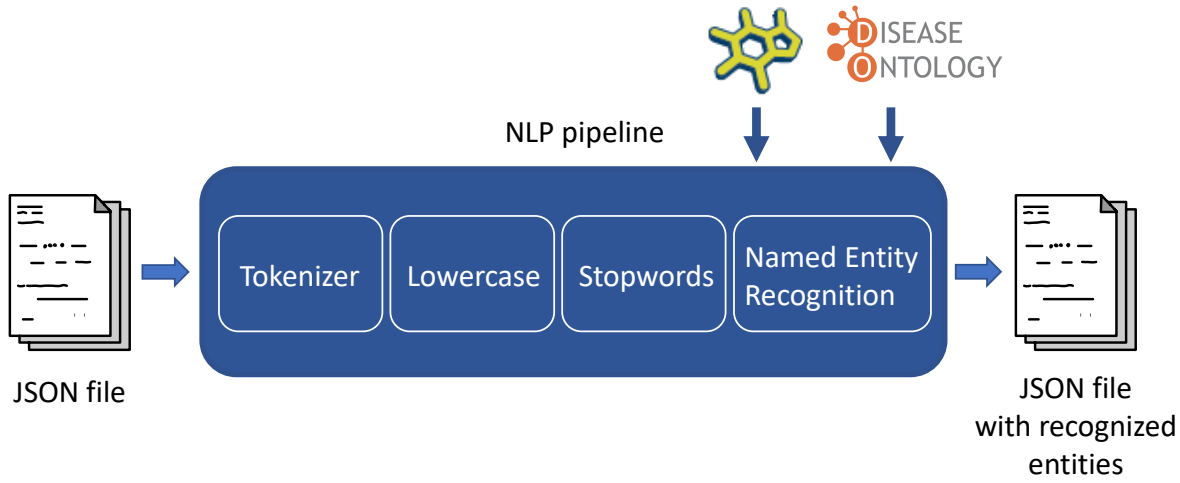


Figure 4.3: Natural Language Processing pipeline architecture

Stopword removal Lowercase tokens that do not have relevance to our problem domain (also known as stopwords [35]) are removed. As these stopwords are removed, the total number of words also decreases, improving the performance and accuracy of subsequent tasks and algorithms. Here, are also removed known words that can interfere with the subsequent task, Named Entity Recognition (NER).

Named Entity Recognition Once the resulting data from previous steps is processed, we can proceed to recognize entities that are present in its *corpus* using a module of NER. These entities can be characterized as a word or series of words that systematically refer to the same thing. This module then seeks to identify this key information (e.g. entities) in unstructured text. For various common use cases, there are NER modules available that can provide accurate results while delivering near-human performance [40]. More technical domains, like the life science industry, require custom entity tagging to address the domain-specific use case. To achieve that, raise the combination of NER modules with the knowledge present in ontologies, which allows for improved entity recognition. The justification for that relies on ontologies to understand the domain-specific entities as concepts, including their context, which makes this NER modules a lot more suitable to identify those entities in text *corpus*. This process is done using MERPy Python module [10] that consists of a NER tool that given any lexicon and an input text, returns a list of terms recognized in the text, including its locations (commonly known as annotations) — Entity Linking. No annotated training data is required to identify new entities. The library can obtain the ontologies that allow the recognition of entities related to active ingredients and therapeutic

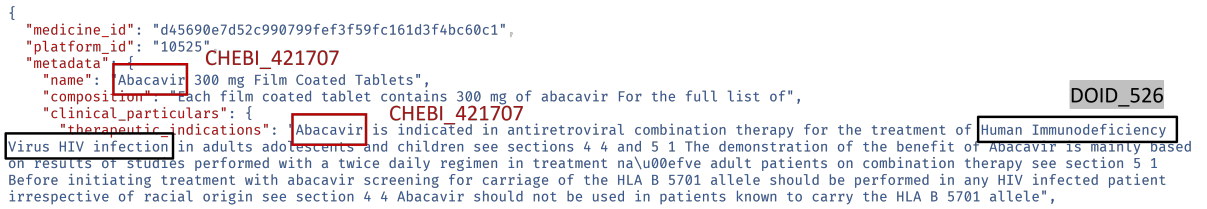


Figure 4.4: An example of an annotated sentence retrieved from Abacavir's PIL. The chemical entities are linked to the ChEBI ontology (CHEBI:421707), and the disease entity is linked to the DO ontology (DOID:526).

indications. We use the ChEBI and DO ontologies to allow active ingredients and disease entity recognition, respectively. Figure 4.4 depicts the annotation of all entities present in the Abacavir's PIL.

We must point out that some fine-tuning was performed in this module. This can result in misunderstandings if synonyms or related words are used instead of the specific disease described in DO. One particular observable case was the use of HIV while DO only specifies the HIV as HIV Infection. As diseases are found not to be detected in these cases, more replacements can be defined, improving the recognition rate. The output of the NER module is a JSON set of files, one for each medicine, with the recognized entities.

This tool obtains itself the necessary ontologies that allow entity recognition both for active ingredients and therapeutic indications, represented by **ChEBI**[7] and **DO**[8] ontologies respectively.

After this task ends, it outputs a new set of JSON files with recognized entities.

4.4 Dataset Generation

After recognizing all the ontology entities on the NER module, we aggregate all JSON file information to create an RS dataset known as Med4EMC and Med4DB, depending on the source, either EMC or DrugBank, in Comma Separated Values (CSV) format. A typical dataset structure for a RS consists of instances of the type: <user, item, rating>. A widespread usage for this format exists in movie recommendation, where the user corresponds to a person, the item to a movie that person watched, and the rating that the person gave to the movie, after watching it.

[7] Available at: https://www.ebi.ac.uk/chebi/
[8] Available at https://disease-ontology.org/

In our particular research, is important to retain a bit more information to allow for further analysis and dataset uses. The Table 4.2 illustrates our dataset structure.

Table 4.2: Dataset instance structure specification, using abacavir as an example. The user represents the disease ontology ID, the item represents the medicine, and the rating corresponds to the rank. The remaining fields are solely used to communicate the recommendation for the end-users

Attribute	Value	Example
user	Disease ontology ID	DOID:526
username	Name of the disease	human immunodeficiency virus infectious disease
item	ChEBI ID	CHEBI:421707
item_name	Name of the chemical entities	abacavir
rating	Implicit rating associated with therapeutic indications	1
date	Timestamp of a rating in the format YYYY-MM-DD	2022-12-01

In this dataset, we grab a couple of the JSON properties like (1) therapeutic indications, and (2) disease to define the rating values, as well as the (3) user, (4) username, (5) item, (6) item_name for each recognized entity inside those fields, and (7) date. The date field will help provide recommendations based on historical data and trends over time in the future. Other fields like incompatibilities could also allow exploration for DDI. Each dataset instance receives an implicit rating based on the item's adequacy to the given user.

After all the data sources (EMC and Drugbank) have been processed, a CSV file is produced to represent them. It should be noted that the datasets have a high level of sparsity, with EMC being around 92% and Drugbank being around 98%.

4.4.1 Dataset Augmentation using Knowledge-Based Similarity

After the NLP pipeline has processed and identified the ontology entities, in step ③ , we apply in step ④ our hypothesis of enriching the generated datasets with closely similar entities, that represent related chemicals to the ones existent in the analyzed medicines. To achieve that, a Knowledge Graph (KG) can be constructed to represent these relationships. This Knowledge Graph (KG) is intended to analyze relationships and allow semantic and structural similarity comparisons between entities and their parents. If the effective similarity between an identified entity and its parents is contained in the first quartile (for instance), the parent can also be considered as an option for use in the RS. Figure 4.5 presents an example of a relation "is-a" for the item tannic-accid (CHEBI:75211) and its parents. This oriented graph representation captures the

relationship between an entity and its parents in the KG.

The augmented datasets, named Med4EMC-KG and Med4DB-KG, are generated by traversing these knowledge graphs for each recognized entity and adding them to the dataset based on the previously explained criteria. Knowledge-based similarity measures the semantic or structural relation degree between two concepts, considering the available knowledge in a particular domain or subject area. It is particularly useful for use in information retrieval, document classification, or in RS.

To calculate knowledge-based similarities, NLP techniques such as natural language parsing, entity recognition, and word embeddings can be used. In our particular case study, we apply three well-known information content-based similarity measures such as: (1) Resnik [58], (2) Jiang and Conrath [34], and (3) Lin [41]. We also apply two well-known structural similarity measures for quantifying molecular similarity, such as: (1) Morgan [48], and (2) Tanimoto [6]. Bajusz et al. [6] draw Tanimoto, Dice, Cosine, and Soergel as the best metrics for similarity calculations. Using other words, they produce the closest rankings to the average composite rankings of the eight distance measures examined in this study (i.e. the Tanimoto index stands out as the best index compared to all remaining ones). To determine the structural similarity between two molecules it is required to have a representation of their structures. ChEBI offers a structure compilation in Simplified Molecular-Input Line-Entry System (SMILES), International Chemical Identifier (InChI) [66], and the MDL chemical file formats. These formats were selected based on preference, with Simplified Molecular-Input Line-Entry System (SMILES) being the preferred one due to its widespread adoption, over the remaining ones [66]. It is important to note that not all chemical entities have all three types of structural representations. DiShIn[9] tool [11] is designed to determine semantic similarities among entities depicted by an ontology.

The Med4EMC-KG and Med4DB-KG datasets have a similar format to the Movielens and Netflix prize, which means they can be read using existing open RS frameworks. Its file structure is formed by tuples detailed in Table 4.2. The final datasets are present in the project's repository, mentioned in Chapter 1. The high sparsity of this dataset with a similar size to the aforementioned datasets is remarkable.

4.5 Recommender System and Evaluation

In this last section, we use all the generated datasets to fuel RSs — Med4EMC, MedDB, Med4EMC-KG, and Med4DB-KG. To have a better comparison, we chose to adopt

[9] https://www.lasige.pt/tools/dishin/

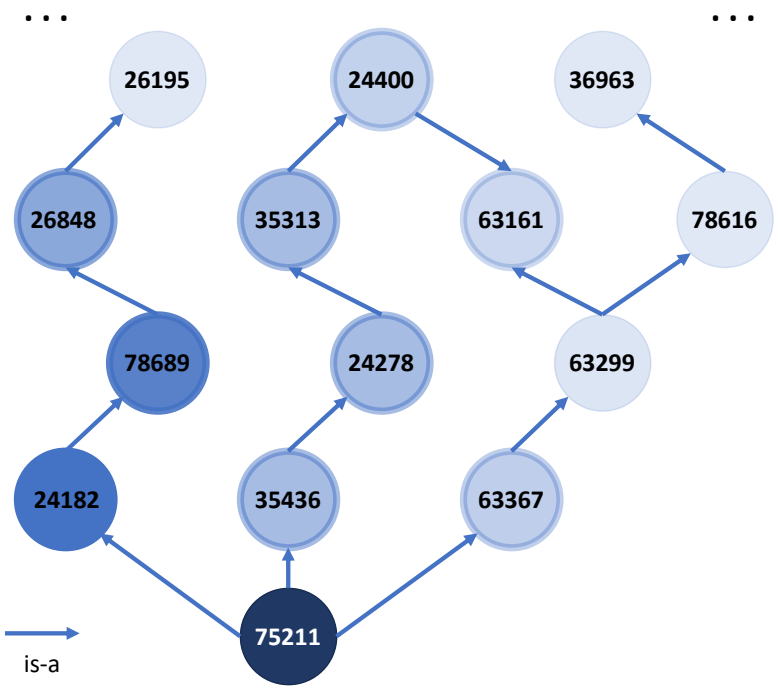


Figure 4.5: An example of a relation "is-a" from the Knowledge graph for the item tannic acid (CHEBI:75211) and its parents. In this example, a graph connection contributes to a recommendation for new datasets — Med4EMC-KG and Med4DB-KG. Saturation represents semantic similarity with a descendant — the root (the darker means more similar); thickness represents set inclusion (greater thickness belongs to the final dataset).

the algorithms implemented by Barros et al.[9]. Alternating Least Squares (ALS) and Bayesian Personalized Ranking (BPR) are used for CF, while the semantic similarity between chemical compounds, called ONTO, is used for CB. In summary, the ONTO algorithm uses two item lists, train and test, as input. The train set contains the user as seen, while the test set contains items to recommend. ONTO then calculates the similarity of each test item to each training item and averages those similarities [9]. The hybrid approach combines both CF and CB algorithm results to provide recommendation results, being said, it combines the scores for ALS and BPR (CF) with the ONTO algorithm scores (CB) [3]. This score fusion uses a weighted approach. The components are weighted using three different metrics. To have a better result comparison, we used the same evaluation procedure done in [9]. The M_1 represents the score multiplication from CF and CB approaches, as in Eq. 4.1. M_2 represents the scores mean for the same approaches, as in Eq. 4.2. M_3 represents the scores quadratic mean for the same approaches, as in Eq. 4.3.

$$M_1 = S_{CF} \times S_{CB} \tag{4.1}$$

$$M_2 = \frac{S_{CF} + S_{CB}}{2} \tag{4.2}$$

$$M_3 = \frac{\sqrt{S_{CF}^2 + S_{CB}^2}}{2} \tag{4.3}$$

Both ALS and BPR use lies in excellent performance shown using implicit feedback datasets, which is the case for the generated datasets. The evaluated algorithms in this paper are summarized in Table 4.3. For collaborative filtering, ALS and BPR were tested separately. Regarding content-based filtering, we assessed the ONTO algorithm using three distinct similarity measures: Lin, Resnik, Jiang and Conrath (JC). Also, hybrid models were created by combining CF and CB methods, utilizing the aforementioned metrics (Eq.s 4.1 to 4.3) to compute each test item's final score.

Table 4.3: List of the algorithms evaluated. More than 23 different algorithms are applied. For instance, ONTO_JC means the ONTO Content-based with Jiang and Conrath measure. Also, ALS_ONTO_JC_1 means Hybrid (ALS plus ONTO) with Jiang and Conrath measure and M_1.

CF	**CB**	**Metric**	**Algorithm**
ALS	–	–	ALS
BPR	–	–	BPR
–	ONTO_[JC|LIN|RESNIK]	–	ONTO_[JC|LIN|RESNIK]
ALS	ONTO_[JC|LIN|RESNIK]	$M[1\|2\|3]$	ALS_ONTO_[JC|LIN|RESNIK]_[1|2|3]
BPR	ONTO_[JC|LIN|RESNIK]	$M[1\|2\|3]$	BPR_ONTO_[JC|LIN|RESNIK]_[1|2|3]

Legend: The square brackets represent an optional part of the statement, and the vertical line describes an "OR" condition.

An item recommendation algorithm's effectiveness is generally assessed through a ranking metric that evaluates how accurately it ranks relevant items at various positions [57]. Offline methods are used due to the availability of pre-existing datasets [25]. Here, two types of metrics are used, Classification Accuracy Metrics (CAMet) and Rank Accuracy Metrics (RAMet). For CAMet we use Precision (Eq. 4.4), Recall (Eq. 4.5), while for RAMet we use Mean Reciprocal Rank (MRR) (Eq. 4.6) and Normalised Discount Cumulative Gain (nDCG) (Eq. 4.8).

$$Precision@k = \frac{relevant\_items@k}{k} \tag{4.4}$$

$$Recall@k = \frac{relevant\_items@k}{all\_relevant\_items} \tag{4.5}$$

$$MRR = \frac{1}{n\_users} \sum_{i=1}^{n\_users} \frac{1}{rank_i} \tag{4.6}$$

$$DCG = \sum_{i=1}^{n} \frac{relevance_i}{log_2(i+1)} \tag{4.7}$$

$$nDCG = \frac{DCG}{iDCG} \tag{4.8}$$

The Precision@k metric provides a measure of items recommended in the top@k list. Meanwhile, Recall@k measures the number of relevant items suggested in the top@k list. The Mean Reciprocal Rank (MRR) determines the position of the first relevant item's appearance. Lastly, the Normalised Discount Cumulative Gain (nDCG) is an evaluation approach that compares the ideal ranking of a test set (iDCG) with the ranking assigned by the recommendation algorithm.

5

Results and Discussion

This section details the results of applying the developed pipeline to EMC and Drugbank use cases. Table 5.1 displays some statistical information referring to the Med4EMC, Med4DB datasets, and the augmented datasets using knowledge-based similarity. The EMC baseline dataset, Med4EMC, has 29 unique users and 26 unique items. For its respective knowledge-based similarity-augmented datasets, we achieved around 135 unique items, representing a sparsity value of 81%, using semantic similarity. Using structural similarity metrics, we reached about 32 unique items, representing a sparsity value of around 84%. The Drugbank baseline dataset, Med4DB, has 734 unique users and 1478 unique items. For its respective knowledge-based similarity-augmented datasets, we reached around 2850 unique items, representing a sparsity value of around 98%, using semantic similarity. Using structural similarity metrics, we reached around 1590 unique items, representing a sparsity value of around 99%.

The top@5 items rated by users are presented in Table 5.2. the most rated item is CHEBI:367163 (duranavir), with ratings from 41 users, and CHEBI:37527 (acid), with 153 user ratings for Med4EMC and Med4DB, respectively. Moreover, for Med4EMC, three diseases have solely one rated item (encountering the cold start problem). For Med4DB, this number is 248. In our context, this means that 10.3% of diseases possess just one therapeutic indication from our EMC list, while it rises to 33.8% for the DrugBank list.

The resume outcomes for Med4EMC, Med4DB, Med4EMC-KG, and Med4DB-KG using Resnik and Tanimoto similarity measures are presented in Table 5.1, focusing on

Table 5.1: Statistics of the Datasets Med4EMC and Med4DB Corpus Regarding Knowledge Graph-Based Recommendation for semantic and structural similarities

	dataset	users	items	ratings	sparsity
Baseline	Med4EMC	29	26	353	91,8%
	Med4DB	734	1478	5534	99,6%
Knowledge graph — semantic	Med4EMC-KG-resnik	29	134	703	81,9%
	Med4EMC-KG-jc		136	784	80,1%
	Med4EMC-KG-lin		137	728	81,7%
	Med4DB-KG-resnik	734	2838	33139	98,3%
	Med4DB-KG-jc		2900	35366	98,3%
	Med4DB-KG-lin		2868	33472	98,4%
Knowledge graph — structural	Med4EMC-KG-morgan	29	32	152	83,6%
	Med4EMC-KG-tanimoto			141	84,8%
	Med4DB-KG-morgan	734	1598	7166	99,3%
	Med4DB-KG-tanimoto		1585	7016	99,3%

Table 5.2: Top@5 items rated by users in the baseline datasets: Med4EMC and Med4DB

EMC			DrugBank		
ID	**name**	**#**	**ID**	**name**	**#**
CHEBI:367163	darunavir	41	CHEBI:37527	acid	153
CHEBI:37924	atazanavir	28	CHEBI:9667	triamcinolone	57
CHEBI:16189	sulfate	26	CHEBI:16189	sulfate	35
CHEBI:63577	lamivudine	23	CHEBI:52999	interferon	34
CHEBI:63613	nevirapine	21	CHEBI:30089	acetate	33

Top@5 recommendations, obtained using the five most similar items when calculating ONTO algorithm scores. The next consideration will take the Med4EMC-KG-resnik dataset as a starting point. Similar observations are made for the remaining datasets, except for Med4EMC. This thesis report includes an appendix with all the resulting heatmaps considering all datasets and covering all remaining algorithms described in Table 4.3. Fig.s 5.1 to 5.4 show the below results using heat maps, helping identify patterns and trends.

The hybrid models using ALS and BPR showed the best results for all evaluated metrics. More specifically, ALS_ONTO_Resnik_1 showed the highest precision and recall values (0.83 and 0.45, respectively), which corresponds to an improvement of 7% and 1% over the ALS results. Greater result differences are obtained using ONTO_Resnik for precision (17%). Worth noting that BPR produced lower results than ALS for all evaluated metrics. However, BPR and ONTO combination, BPR_ONTO, produced an

improvement in precision and recall of approximately 25% and 35%, respectively — a more significant improvement than the increase from ALS to ALS_ONTO. The results highlight the effectiveness of combining ALS with ONTO to achieve the highest performance levels. However, the BPR hybrids experienced more substantial improvements compared to standalone BPR. These improvements in precision and recall metrics indicate that hybrid algorithms are the most successful in including relevant items in the recommended list.

When evaluating the ranking quality metrics for Mean Reciprocal Rank (MRR) and Normalised Discount Cumulative Gain (nDCG), ALS_ONTO_Resnik_1 outperformed the other models, achieving values of 0.98 and 0.97, respectively. This represents an improvement of 17% and 9%, respectively, over the ALS results alone. MRR increased by 12% for BPR, while nDCG showed an increase of 9%. These results help highlight the hybrid algorithms' effectiveness in improving recommendations' ranking.

When examined in isolation, the ONTO algorithm produced the lowest results across all metrics, except over EMC datasets. However, they followed a similar trend to all remaining algorithms. In particular, ONTO benefits from being a CB algorithm, which mitigates the challenges of the cold start problem, where it is not affected over the EMC datasets (remember that 10.3% of diseases have only a single therapeutic indication). Unlike ALS and BPR, which require items in the training set, ONTO requires all entities to be present in an ontology. Consequently, chemical compounds must be represented in ChEBI. When assessing the BPR, we found that the hybrid with ONTO_Resnik and M_1 gave comparable results to those with ONTO_Lin and ONTO_JC. Interestingly, the hybrid ONTO_Resnik outperformed the alternatives with M_2. It is worth noting that BPR inherently increases all scores by one, resulting in scores above one for all items within this algorithm.

When comparing ALS and BPR, ALS showed the highest results, as BPR was explicitly designed to rank, we expected better results. However, the dataset's characteristic of having numerous items of similar relevance contributes to this result — resulting in challenging ranking scenarios.

It is worth mentioning, that across all evaluation metrics, quality metrics decreased when the value of "k" increased, except for the recall metric. Hybrid algorithms' quality metrics are also affected as the value of "k" increases. These results are consistent with Barros et al. [9] conclusions and confirm our work's validity. This is a first step towards our methodology, finding new recommendation algorithms and exploring new concepts in content-based recommender systems.

Table 5.3: Final results for Med4EMG-KG and Med4DB-KG with Resnik and Tanimoto similarity measures, considering Top@5 recommendations. P stands for Precision, and R for Recall.

Dataset	Algorithm	P@5	R@5	MRR@5	nDCG@5
Med4EMC-KG-Resnik	ALS	0.77	0.42	0.81	0.88
	BPR	0.50	0.22	0.71	0.75
	ALS_ONTO_Resnik_1	0.83	0.45	0.98	0.97
	BPR_ONTO_Resnik_1	0.67	0.30	0.81	0.82
	ONTO_Resnik	0.68	0.36	0.89	0.92
Med4EMC-KG-Tanimoto	ALS	0.47	0.76	0.95	0.96
	BPR	0.30	0.48	0.61	0.60
	ALS_ONTO_Resnik_1	0.44	0.73	0.95	0.93
	BPR_ONTO_Resnik_1	0.31	0.47	0.62	0.60
	ONTO_Resnik	0.49	0.62	0.97	0.96
Med4DB-KG-Resnik	ALS	0.66	0.35	0.90	0.90
	BPR	0.23	0.36	0.38	0.42
	ALS_ONTO_Resnik_1	0.64	0.31	0.87	0.89
	BPR_ONTO_Resnik_1	0.36	0.19	0.60	0.65
	ONTO_Resnik	0.5	0.23	0.79	0.80
Med4DB-KG-Tanimoto	ALS	0.19	0.20	0.38	0.42
	BPR	0.17	0.14	0.31	0.35
	ALS_ONTO_Resnik_1	0.25	0.21	0.45	0.47
	BPR_ONTO_Resnik_1	0.27	0.21	0.46	0.48
	ONTO_Resnik	0.20	0.23	0.45	0.47

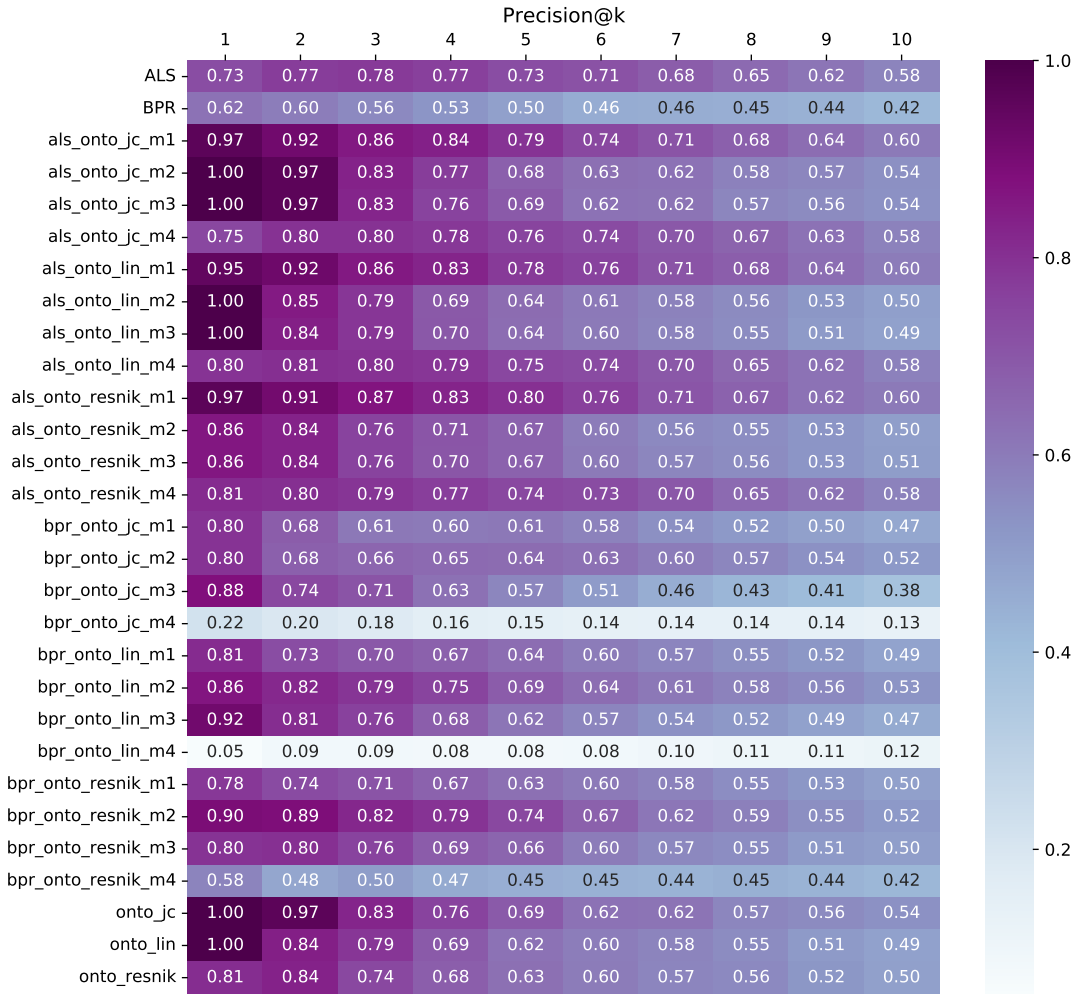


Figure 5.1: A list of top@10 Precision results from Med4EMC-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

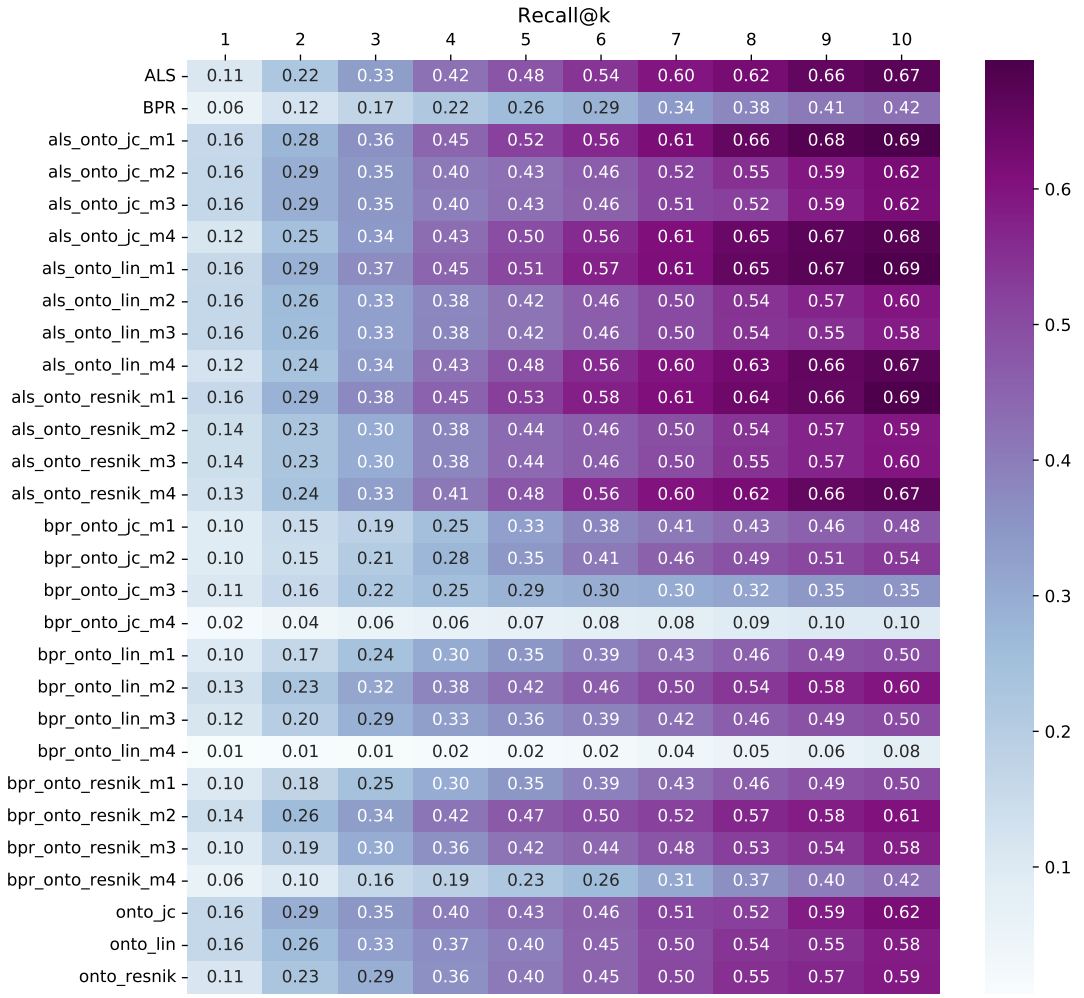


Figure 5.2: A list of top@10 Recall results from Med4EMC-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

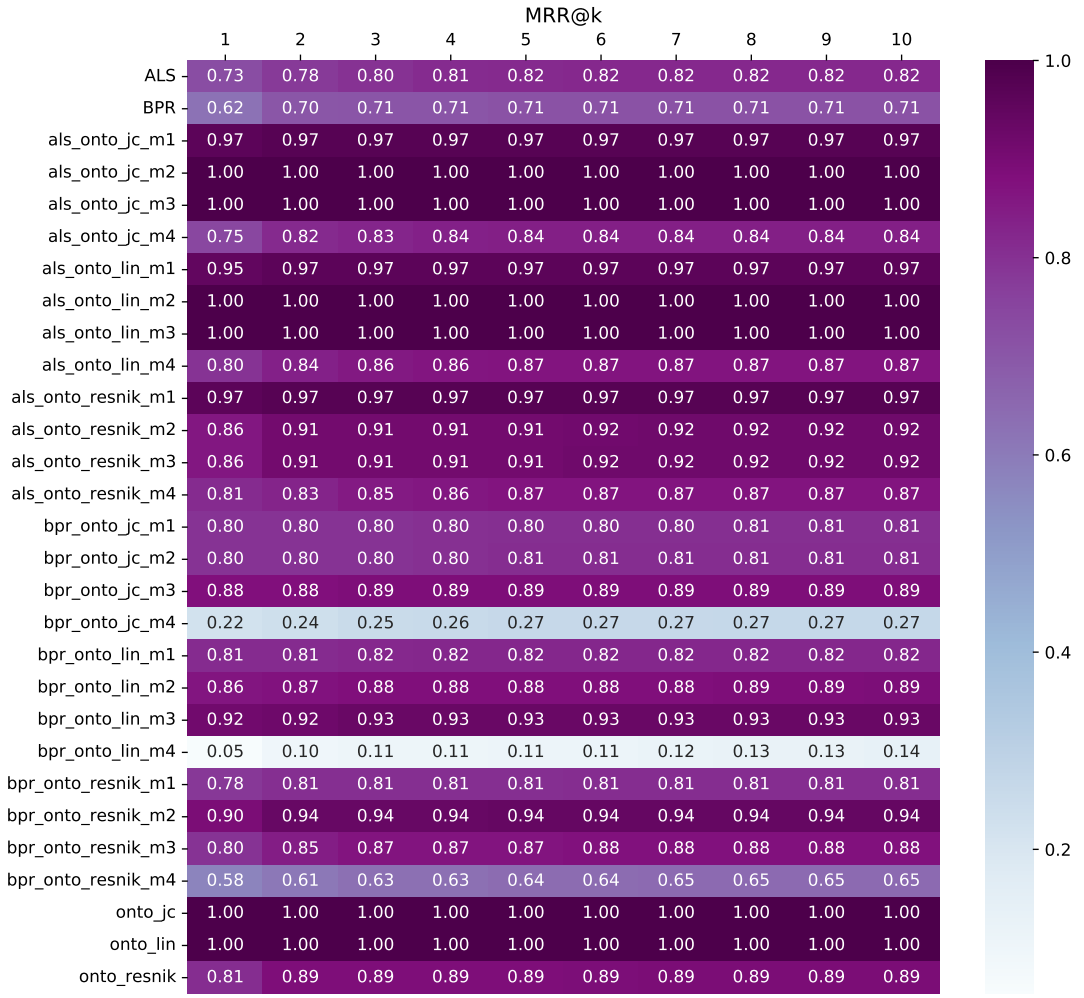


Figure 5.3: A list of top@10 MRR results from Med4EMC-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

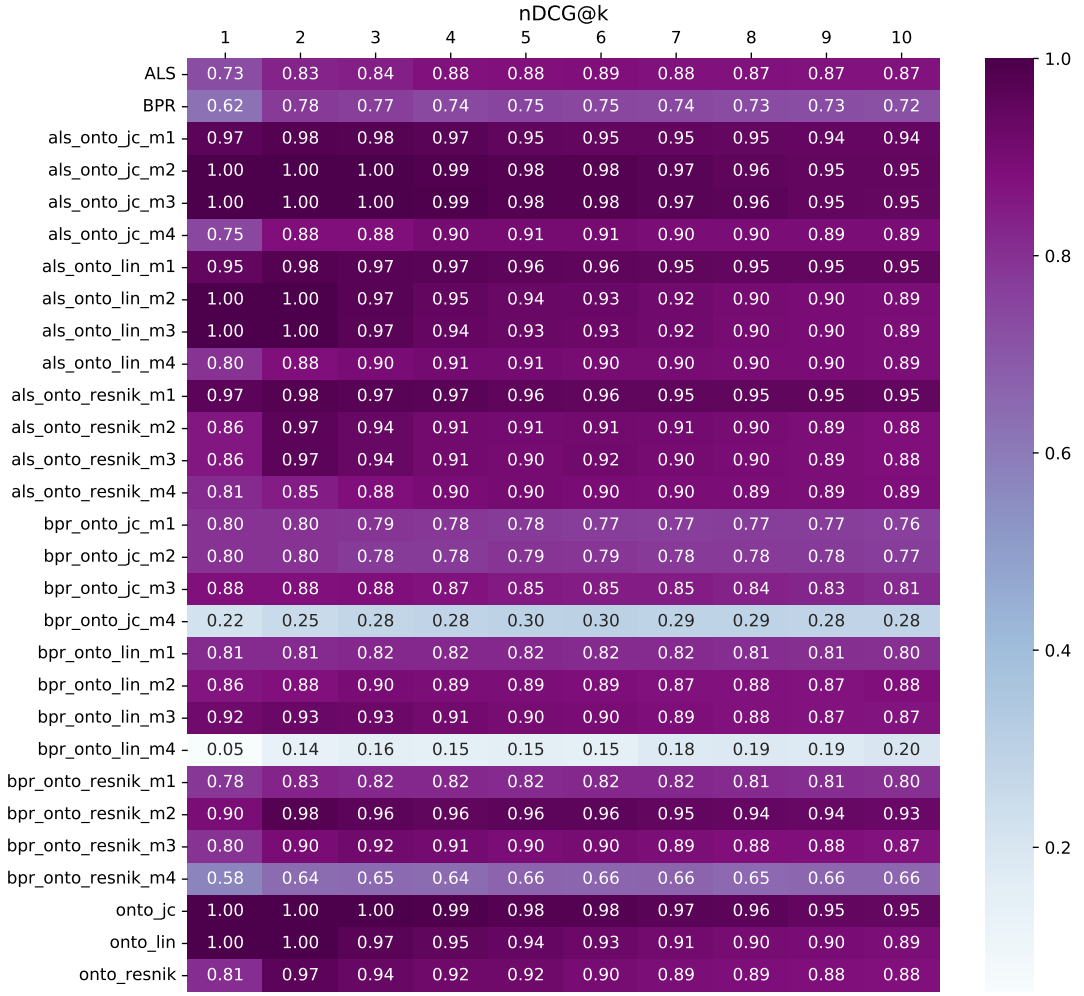


Figure 5.4: A list of top@10 nDCG results from Med4EMC-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

6

Conclusions

With the intent of exploring our hypothesis of processing PIL information to extract valuable and usable data from it to fuel a dataset generation pipeline that can be used to power RSs. Those RSs can then help in the decision-making process portraited by health professionals to better, and more adequately prescribe medicines to their patients. This thesis report demonstrates the achieved process and its results reveal that PIL information processing can have a great impact on the medicine prescription field.

One of the limitations present in this case study lies in the available data volume, as EMC greatly restricted the web scrapping process after we attempted to partner with them.

As future work, it is seen as important and relevant to partner with entities like EMC or DrugBank to increase data quality and volume to improve results. Another important direction to consider is implementing Morgan and Tanimoto as ONTO algorithms, similar to what has been done to Lin, Resnik, and JC. Our approach could also be expanded by considering other types of algorithms and considering a binary problem scenario, i.e. considering therapeutic indications and contraindications.

One of the research directions may collide in the use of patient clinical data to better exclude previously taken medicines to improve DDIs detection and disease/medicine incompatibilities.

References

[1] Yanchao Tan, Chengjun Kong, Leisheng Yu, Pan Li, Chaochao Chen, Xiaolin Zheng, Vicki S. Hertzberg, and Carl Yang, "4sdrug: Symptom-based set-to-set small and safe drug recommendation", in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ser. KDD '22, Washington DC, USA: Association for Computing Machinery, 2022, 3970–3980, ISBN: 9781450393850. DOI: 10.1145/3534678.3539089. [Online]. Available: https://doi.org/10.1145/3534678.3539089.

[2] Rishabh Mittal and Anchal Garg, "Text extraction using ocr: A systematic review", in *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, IEEE, 2020, pages 357–362. DOI: 10.1109/ICIRCA48905.2020.9183326.

[3] Charu C Aggarwal and Charu C Aggarwal, "Ensemble-based and hybrid recommender systems", *Recommender Systems: The Textbook*, pages 199–224, 2016. DOI: 10.1007/978-3-319-29659-3_6.

[4] Xavier Amatriain and Justin Basilico, "Recommender systems in industry: A netflix case study", in *Recommender Systems Handbook*, Francesco Ricci, Lior Rokach, and Bracha Shapira, Eds. Boston, MA: Springer US, 2015, pages 385–419, ISBN: 978-1-4899-7637-6. DOI: 10.1007/978-1-4899-7637-6_11. [Online]. Available: https://doi.org/10.1007/978-1-4899-7637-6_11.

[5] Maryam Habibi, Leon Weber, Mariana Neves, David Luis Wiegandt, and Ulf Leser, "Deep learning with word embeddings improves biomedical named entity recognition", *Bioinformatics*, vol. 33, no. 14, pages i37–i48, Jul. 2017, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btx228. eprint: https://academic.oup.com/bioinformatics/article-pdf/33/14/i37/

50314882/bioinformatics_33_14_i37.pdf. [Online]. Available: https://doi.org/10.1093/bioinformatics/btx228.

[6] Dávid Bajusz, Anita Rácz, and Károly Héberger, "Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations?", *Journal of cheminformatics*, vol. 7, no. 1, pages 1–13, 2015. DOI: 10.1186/s13321-015-0069-3.

[7] Márcia Barros, André Moitinho, and Francisco M Couto, "Using research literature to generate datasets of implicit feedback for recommending scientific items", *IEEE Access*, vol. 7, pages 176 668–176 680, 2019.

[8] Marcia Barros, Francisco M Couto, Matilde Pato, and Pedro Ruas, "Creating recommender systems datasets in scientific fields", in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pages 4029–4030.

[9] Marcia Barros, Andre Moitinho, and Francisco M Couto, "Hybrid semantic recommender system for chemical compounds in large-scale datasets", *Journal of cheminformatics*, vol. 13, no. 1, pages 1–18, 2021.

[10] Francisco M Couto and Andre Lamurias, "Mer: A shell script and annotation server for minimal named entity recognition and linking", *Journal of cheminformatics*, vol. 10, pages 1–10, 2018. DOI: 10.1186/s13321-018-0312-9.

[11] F. Couto and A. Lamurias, "Semantic similarity definition", in *Encyclopedia of Bioinformatics and Computational Biology*, S. Ranganathan, K. Nakai, C. Schönbach, and M. Gribskov, Eds., Oxford: Elsevier, 2019, pages 870–876. DOI: 10.1016/B978-0-12-809633-8.20401-9.

[12] Leilei Sun, Chuanren Liu, Chonghui Guo, Hui Xiong, and Yanming Xie, "Data-driven automatic treatment regimen development and recommendation", Aug. 2016, pages 1865–1874. DOI: 10.1145/2939672.2939866.

[13] Marinka Zitnik, Monica Agrawal, and Jure Leskovec, "Modeling polypharmacy side effects with graph convolutional networks", *Bioinformatics*, vol. 34, no. 13, pages i457–i466, Jun. 2018, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bty294. [Online]. Available: https://doi.org/10.1093/bioinformatics/bty294.

[14] The European Medicines Agency, "Human medicines highlights 2022", Tech. Rep., 2022. [Online]. Available: https://www.ema.europa.eu/en/documents/report/human-medicines-highlights-2022_en.pdf.

[15] The European Medicines Agency, "Guideline on the readability of the labelling and package leaflet of medicinal products for human use", Tech. Rep., 2009. [Online]. Available: https://ec.europa.eu/health/system/files/2016-11/2009_01_12_readability_guideline_final_en_0.pdf.

[16] The European Medicines Agency, "A guideline on summary of product characteristics", Tech. Rep., 2009. [Online]. Available: https://health.ec.europa.eu/system/files/2016-11/smpc_guideline_rev2_en_0.pdf.

[17] Huilian Fan, Yuanchang Zhong, Guangpu Zeng, and Chenhao Ge, "Improving recommender system via knowledge graph based exploring user preference", *Applied Intelligence*, pages 1–13, 2022.

[18] The Food and Drug Administration, "Advancing health through innovation: New drug therapy approvals", Tech. Rep., 2022. [Online]. Available: https://www.fda.gov/drugs/new-drugs-fda-cders-new-molecular-entities-and-new-therapeutic-biological-products/novel-drug-approvals-2022.

[19] Jana Fieselmann, Kübra Annac, Fabian Erdsiek, Yüce Yilmaz-Aslan, and Patrick Brzoska, "What are the reasons for refusing a covid-19 vaccine? a qualitative analysis of social media in germany", *BMC Public Health*, vol. 22, no. 1, pages 1–8, 2022.

[20] Charalampos Doulaverakis, George Nikolaidis, Athanasios Kleontas, and Ioannis Kompatsiaris, "Galenowl: Ontology-based drug recommendations discovery", *Journal of biomedical semantics*, vol. 3, page 14, Dec. 2012. DOI: 10.1186/2041-1480-3-14.

[21] Junyuan Shang, Cao Xiao, Tengfei Ma, Hongyan Li, and J. Sun, "Gamenet: Graph augmented memory networks for recommending medication combination", *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pages 1126–1133, Jul. 2019. DOI: 10.1609/aaai.v33i01.33011126.

[22] Satvik Garg, "Drug recommendation system based on sentiment analysis of drug reviews using machine learning", in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 2021, pages 175–181. DOI: 10.1109/Confluence51648.2021.9377188.

[23] Audrey L Gassman, Christine P Nguyen, and Hylton V Joffe, "Fda regulation of prescription drugs", *New England Journal of Medicine*, vol. 376, no. 7, pages 674–682, 2017.

[24] Lyndsey Elaine Gates and Ahmed Abdeen Hamed, "The anatomy of the sars-cov-2 biomedical literature: Introducing the covidx network algorithm for drug repurposing recommendation", *Journal of medical Internet research*, vol. 22, no. 8, e21169, 2020.

[25] Asela Gunawardana, Guy Shani, and Sivan Yogev, "Evaluating recommender systems", in *Recommender systems handbook*, Springer, 2022, pages 547–601. DOI: 10.1007/978-1-0716-2197-4_15.

[26] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He, "A survey on knowledge graph-based recommender systems", *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pages 3549–3568, 2020.

[27] F Maxwell Harper and Joseph A Konstan, "The movielens datasets: History and context", *ACM transactions on interactive intelligent systems (TIIS)*, vol. 5, no. 4, pages 1–19, 2015.

[28] Alexander Hodkinson, Natasha Tyler, Darren M Ashcroft, Richard N Keers, Kanza Khan, Denham Phipps, Aseel Abuzour, Peter Bower, Anthony Avery, Stephen Campbell, *et al.*, "Preventable medication harm across health care settings: A systematic review and meta-analysis", *BMC medicine*, vol. 18, no. 1, pages 1–13, 2020.

[29] Andreas Holzinger, Matthias Dehmer, Frank Emmert-Streib, Rita Cucchiara, Isabelle Augenstein, Javier Del Ser, Wojciech Samek, Igor Jurisica, and Natalia Díaz-Rodríguez, "Information fusion as an integrative cross-cutting enabler to achieve robust, explainable, and trustworthy medical artificial intelligence", *Information Fusion*, vol. 79, pages 263–278, 2022, ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2021.10.007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253521002050.

[30] Md. Deloar Hossain, Md. Shafiul Azam, Md Jahan Ali, and Hakilo Sabit, "Drugs rating generation and recommendation from sentiment analysis of drug reviews using machine learning", in *2020 Emerging Technology in Computing, Communication and Electronics (ETCCE)*, 2020, pages 1–6. DOI: 10.1109/ETCCE51779.2020.9350868.

[31] Yifan Hu, Yehuda Koren, and Chris Volinsky, "Collaborative filtering for implicit feedback datasets", in *2008 Eighth IEEE International Conference on Data Mining*, IEEE, 2008, pages 263–272.

[32] Tongtong Huang, Linda T Li, Elmer V Bernstam, and Xiaoqian Jiang, "Confidence-based laboratory test reduction recommendation algorithm", *BMC Medical Informatics and Decision Making*, vol. 23, no. 1, pages 1–15, 2023.

[33] Qian Zhang, Guangquan Zhang, Jie Lu, and Dianshuang Wu, "A framework of hybrid recommender system for personalized clinical prescription", in *2015 10th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, 2015, pages 189–195. DOI: 10.1109/ISKE.2015.98.

[34] Jay J Jiang and David W Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy", *arXiv preprint cmp-lg/9709008*, 1997. DOI: 10.48550/arXiv.cmp-lg/9709008.

[35] Jashanjot Kaur and P Kaur Buttar, "A systematic review on stopword removal algorithms", *International Journal on Future Revolution in Computer Science & Communication Engineering*, vol. 4, no. 4, pages 207–210, 2018.

[36] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh, "Natural language processing: State of the art, current trends and challenges", *Multimedia tools and applications*, vol. 82, no. 3, pages 3713–3744, 2023.

[37] Rayan Krishnan, Pranav Rajpurkar, and Eric J Topol, "Self-supervised learning in medicine and healthcare", *Nature Biomedical Engineering*, vol. 6, no. 12, pages 1346–1352, 2022.

[38] Manika Lamba and Margam Madhusudhan, "Exploring ocr errors in full-text large documents: A study of lis theses and dissertations", 2023.

[39] Andre Lamurias, Tiago Grego, and Francisco M Couto, "Chemical compound and drug name recognition using crfs and semantic similarity based on chebi", in *BioCreative Challenge Evaluation Workshop*, vol. 2, 2013, page 75.

[40] Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li, "A survey on deep learning for named entity recognition", *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pages 50–70, 2020. DOI: 10.1109/TKDE.2020.2981314.

[41] Dekang Lin *et al.*, "An information-theoretic definition of similarity", in *Icml*, vol. 98, 1998, pages 296–304.

[42] Manuel Lobo, Andre Lamurias, Francisco M Couto, *et al.*, "Identifying human phenotype terms by combining machine learning and validation rules", *BioMed Research International*, vol. 2017, 2017.

[43] Nidhi Kushwaha, Raman Goyal, Pramiti Goel, Sidharth Singla, and O. Vyas, "Lod cloud mining for prognosis model(case study: Native app for drug recommender system)", *Advances in Internet of Things*, vol. 04, pages 20–28, Jan. 2014. DOI: 10.4236/ait.2014.43004.

[44] Julian N Marewski and Gerd Gigerenzer, "Heuristic decision making in medicine", *Dialogues in clinical neuroscience*, 2022.

[45] Xi Sheryl Zhang, Fengyi Tang, Hiroko H. Dodge, Jiayu Zhou, and Fei Wang, "Metapred: Meta-learning for clinical risk prediction with limited patient electronic health records", ser. KDD '19, Anchorage, AK, USA: Association for Computing Machinery, 2019, ISBN: 9781450362016. DOI: 10.1145/3292500.3330779. [Online]. Available: https://doi.org/10.1145/3292500.3330779.

[46] Chaoqi Yang, Cao Xiao, Lucas Glass, and J. Sun, "Change matters: Medication change prediction with recurrent residual networks", Aug. 2021, pages 3728–3734. DOI: 10.24963/ijcai.2021/513.

[47] Alistair Johnson, Tom Pollard, Lu Shen, Li-wei Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Celi, and Roger Mark, "Mimic-iii, a freely accessible critical care database", *Scientific Data*, vol. 3, page 160 035, May 2016. DOI: 10.1038/sdata.2016.35.

[48] Harry L Morgan, "The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service", *Journal of chemical documentation*, vol. 5, no. 2, pages 107–113, 1965. DOI: 10.1021/c160017a018.

[49] Douglas W Oard and Jinmook Kim, "Implicit feedback for recommender systems", in *15th National Conference on Artificial Intelligence (AAAI'98)*, AAAI Press, 1998, pages 81–83.

[50] Theresa Olubukola Omodunbi, Grace Egbi Alilu, and Rhoda Nsikanabasi Ikono, "Drug recommender systems: A review of state-of-the-art algorithms", in *2022 5th Information Technology for Education and Development (ITED)*, 2022, pages 1–8. DOI: 10.1109/ITED56637.2022.10051591.

[51] Fernando Ortega, Jesús Bobadilla, Abraham Gutiérrez, Remigio Hurtado, and Xin Li, "Artificial intelligence scientific documentation dataset for recommender systems", *IEEE Access*, vol. 6, pages 48 543–48 555, 2018.

[52] Charalampos Doulaverakis, George Nikolaidis, Athanasios Kleontas, and Ioannis Kompatsiaris, "Panacea, a semantic-enabled drug recommendations discovery framework.", *Journal of biomedical semantics*, vol. 5, page 13, Mar. 2014. DOI: 10.1186/2041-1480-5-13.

[53] István Pilászy, Dávid Zibriczky, and Domonkos Tikk, "Fast als-based matrix factorization for explicit and implicit feedback datasets", in *Proceedings of the fourth ACM conference on Recommender systems*, 2010, pages 71–78.

[54] Cartic Ramakrishnan, Abhishek Patnia, Eduard Hovy, and Gully APC Burns, "Layout-aware text extraction from full-text pdf of scientific articles", *Source code for biology and medicine*, vol. 7, pages 1–10, 2012.

[55] Md. Deloar Hossain, Md. Shafiul Azam, Md Jahan Ali, and Hakilo Sabit, "Drugs rating generation and recommendation from sentiment analysis of drug reviews using machine learning", in *2020 Emerging Technology in Computing, Communication and Electronics (ETCCE)*, 2020, pages 1–6. DOI: 10.1109/ETCCE51779.2020.9350868.

[56] Satvik Garg, "Drug recommendation system based on sentiment analysis of drug reviews using machine learning", in *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 2021, pages 175–181. DOI: 10.1109/Confluence51648.2021.9377188.

[57] Steffen Rendle, "Item recommendation from implicit feedback", in *Recommender Systems Handbook*, Springer, 2022, pages 143–171. DOI: 10.1007/978-1-0716-2197-4_15.

[58] Philip Resnik, "Using information content to evaluate semantic similarity in a taxonomy", *arXiv preprint cmp-lg/9511007*, 1995. DOI: 10.48550/arXiv.cmp-lg/9511007.

[59] Francesco Ricci, Lior Rokach, and Bracha Shapira, "Recommender systems: Introduction and challenges", *Recommender systems handbook*, pages 1–34, 2015.

[60] Chaoqi Yang, Cao Xiao, Fenglong Ma, Lucas Glass, and J. Sun, "Safedrug: Dual molecular graph encoders for recommending effective and safe drug combinations", Aug. 2021, pages 3735–3741. DOI: 10.24963/ijcai.2021/514.

[61] Alejandro Rodríguez, Enrique Jiménez, Jesús Fernández, Martin Eccius, Juan Miguel Gómez, Giner Alor-Hernandez, Rubén Posada-Gomez, and Carlos Laufer, "Semmed: Applying semantic web to medical recommendation systems", in *2009 First International Conference on Intensive Applications and Services*, 2009, pages 47–52. DOI: 10.1109/INTENSIVE.2009.12.

[62] Benjamin Stark, Constanze Knahl, Mert Aydin, and Karim Elish, "A literature review on medicine recommender systems", *International journal of advanced computer science and applications*, vol. 10, no. 8, 2019.

[63] Gábor Takács and Domonkos Tikk, "Alternating least squares for personalized ranking", in *Proceedings of the sixth ACM conference on Recommender systems*, 2012, pages 83–90.

[64] Thi Ngoc Trang Tran, Alexander Felfernig, Christoph Trattner, and Andreas Holzinger, "Recommender systems in the healthcare domain: State-of-the-art and research issues", *Journal of Intelligent Information Systems*, vol. 57, pages 171–201, 2021.

[65] World Health Organization, "Patient safety", Tech. Rep., 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/patient-safety.

[66] Daniel S Wigh, Jonathan M Goodman, and Alexei A Lapkin, "A review of molecular representation in the age of machine learning", *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 12, no. 5, e1603, 2022. DOI: 10.1002/wcms.1603.

[67] Rui Wu, Zhaopeng Qiu, Jiacheng Jiang, Guilin Qi, and Xian Wu, "Conditional generation net for medication recommendation", *Proceedings of the ACM Web Conference 2022*, 2022.

[68] Shaojun Zhao, "Named entity recognition in biomedical texts using an hmm model", in *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, ser. JNLPBA '04, Geneva, Switzerland: Association for Computational Linguistics, 2004, 84–87.

Heatmap Results

These heatmaps offer valuable insights into the effectiveness of the algorithms by showcasing their respective strengths and weaknesses.

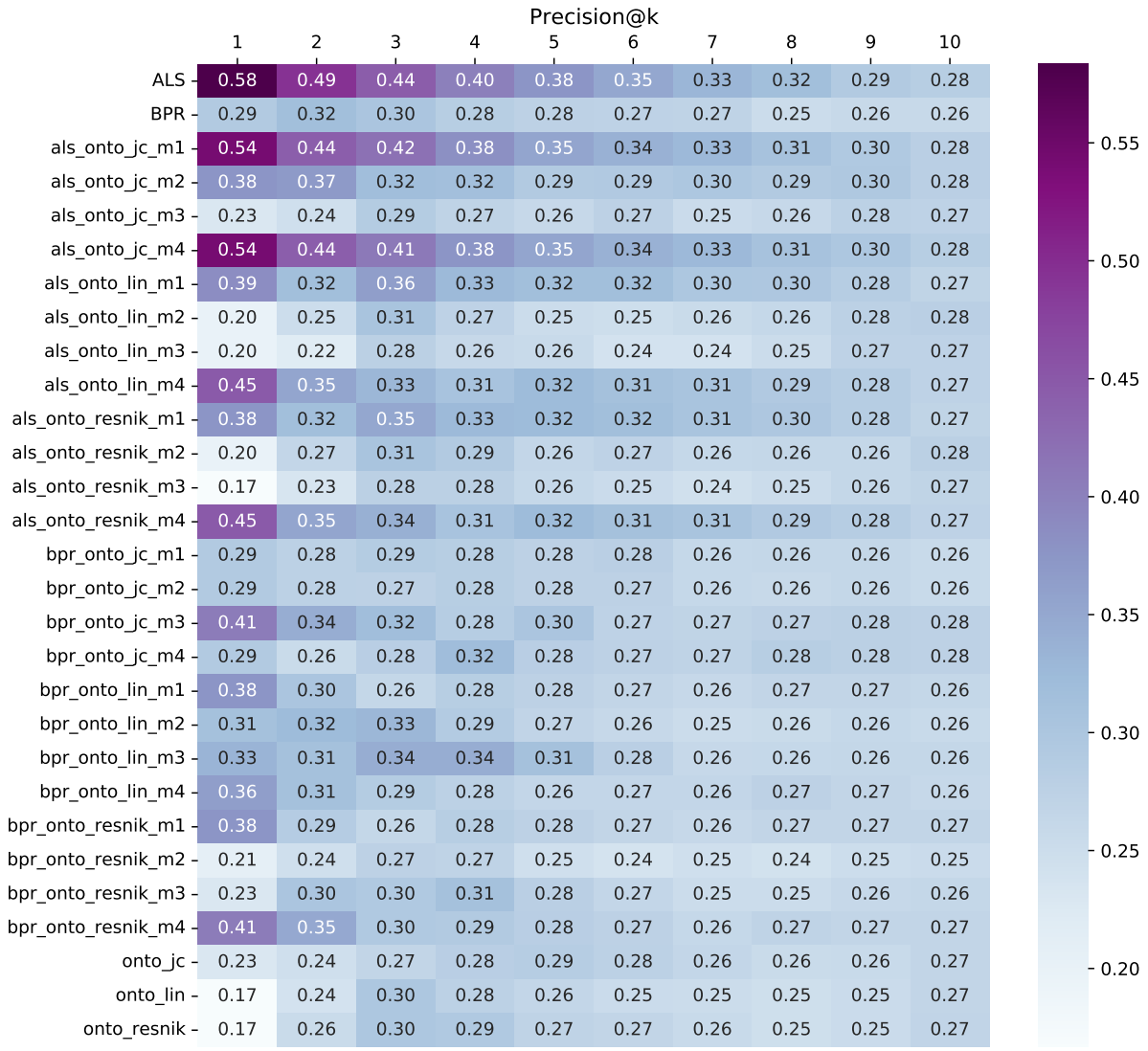


Figure A.1: A list of top@10 Precision results from Med4EMC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

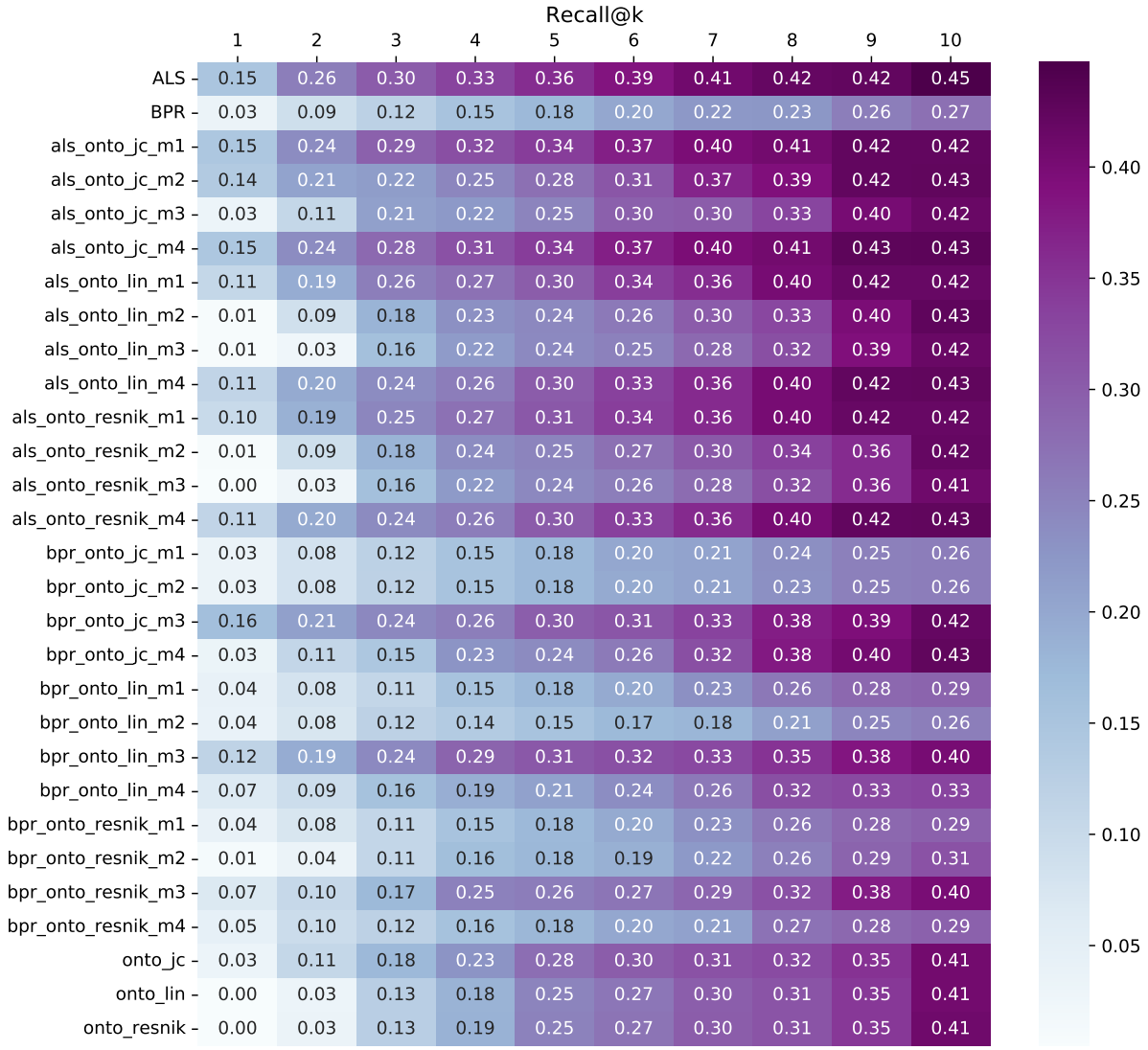


Figure A.2: A list of top@10 Recall results from Med4EMC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

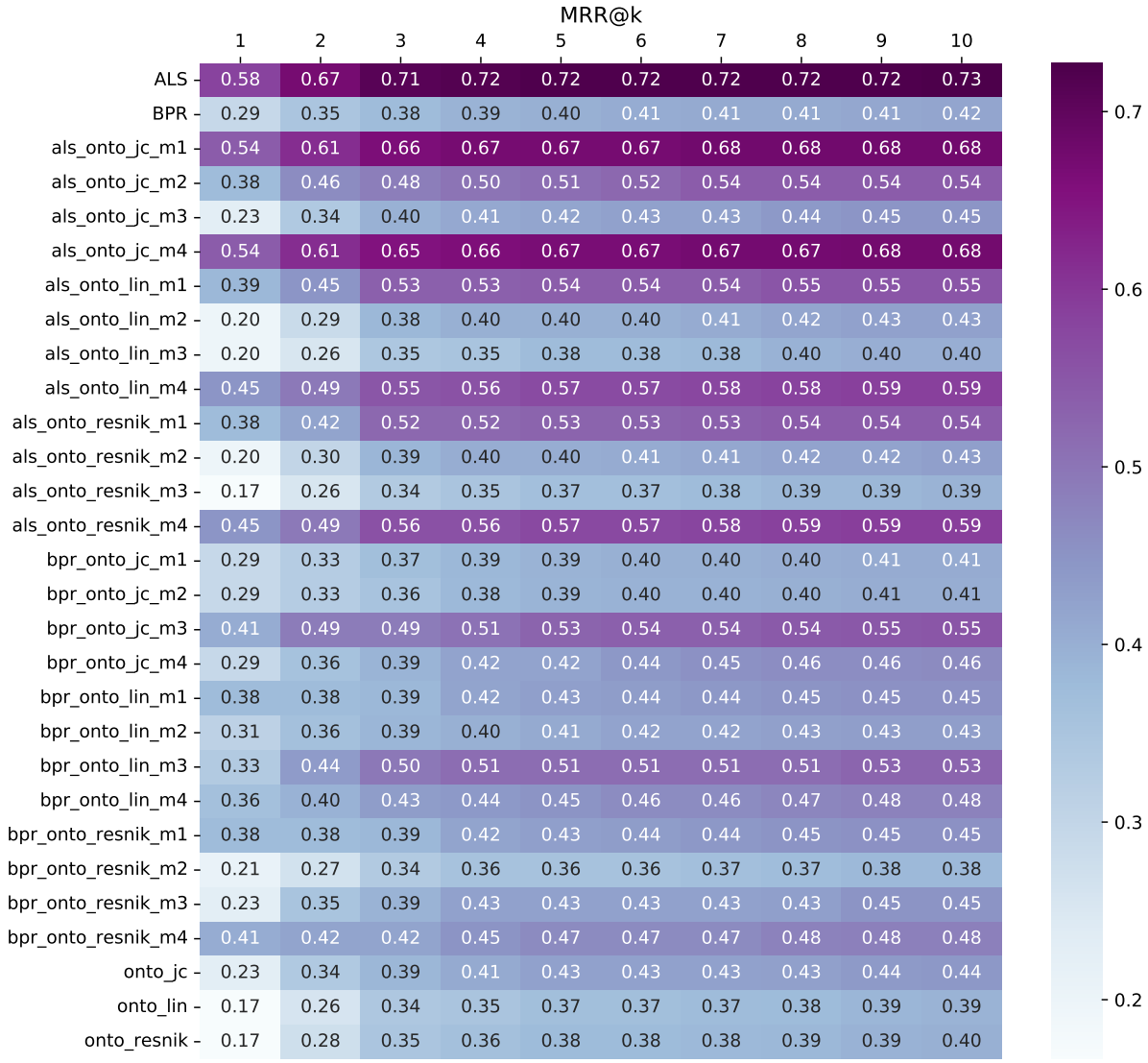


Figure A.3: A list of top@10 MRR results from Med4EMC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

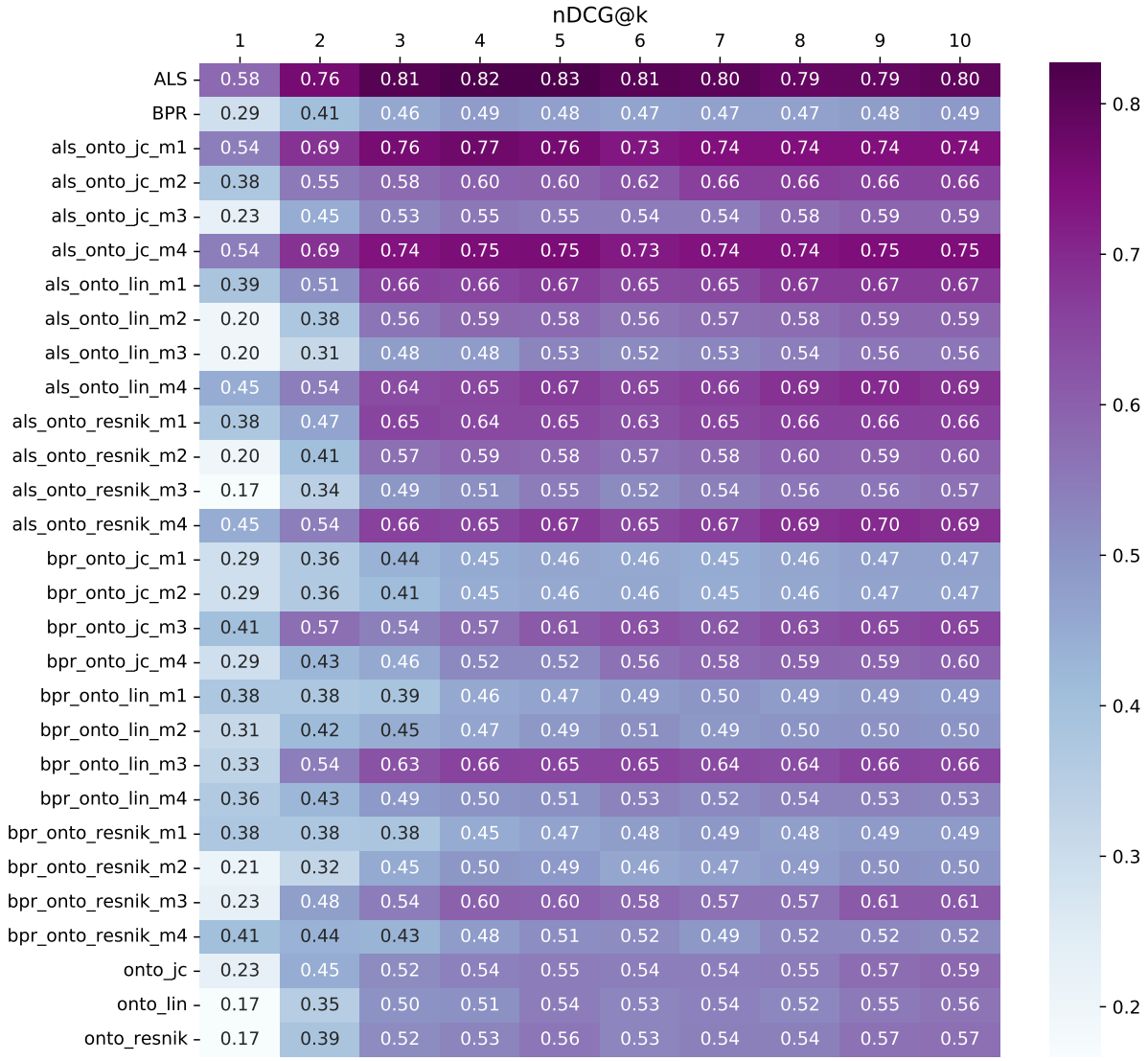


Figure A.4: A list of top@10 nDCG results from Med4EMC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

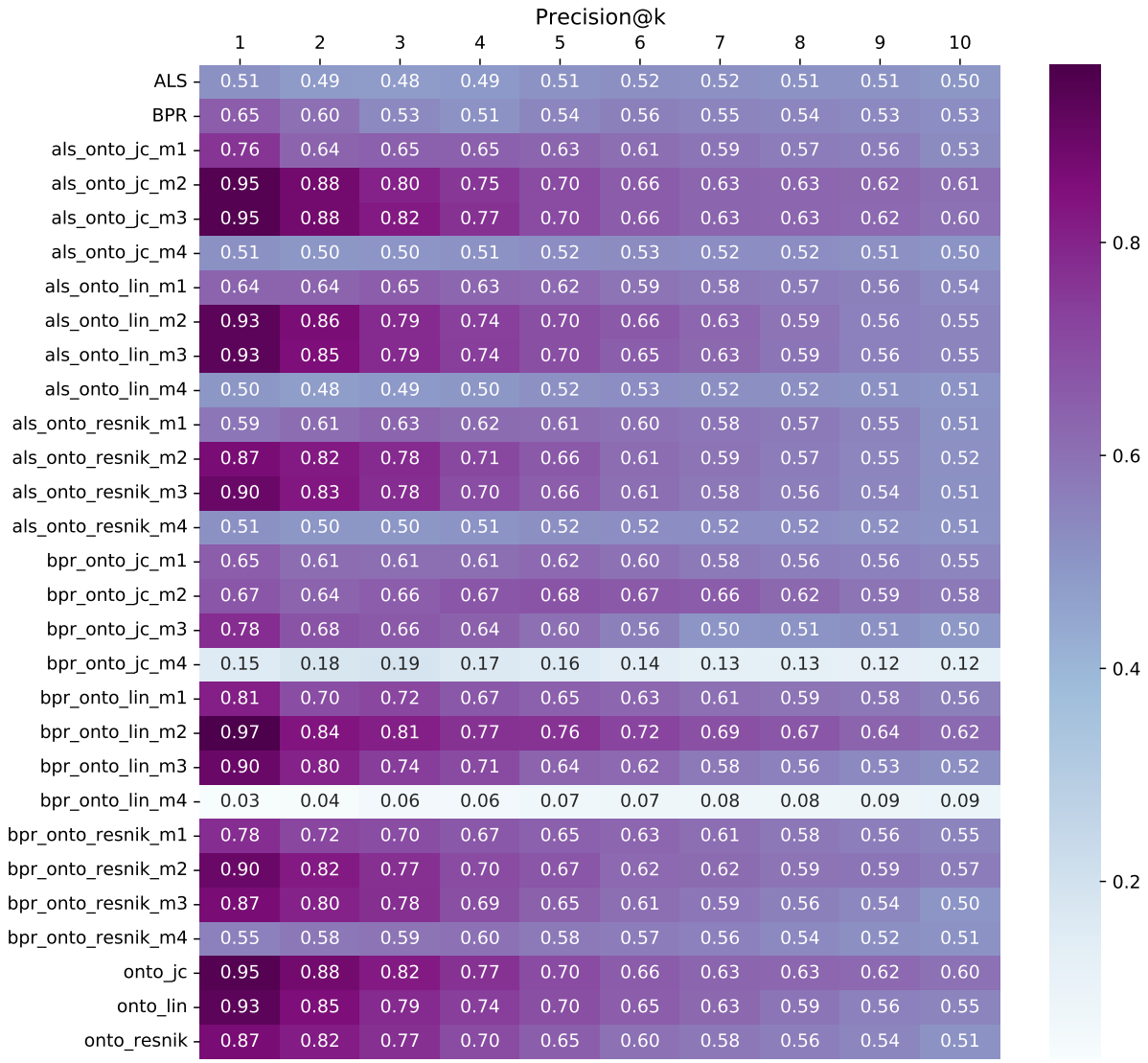


Figure A.5: A list of top@10 Precision results from Med4EMC-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

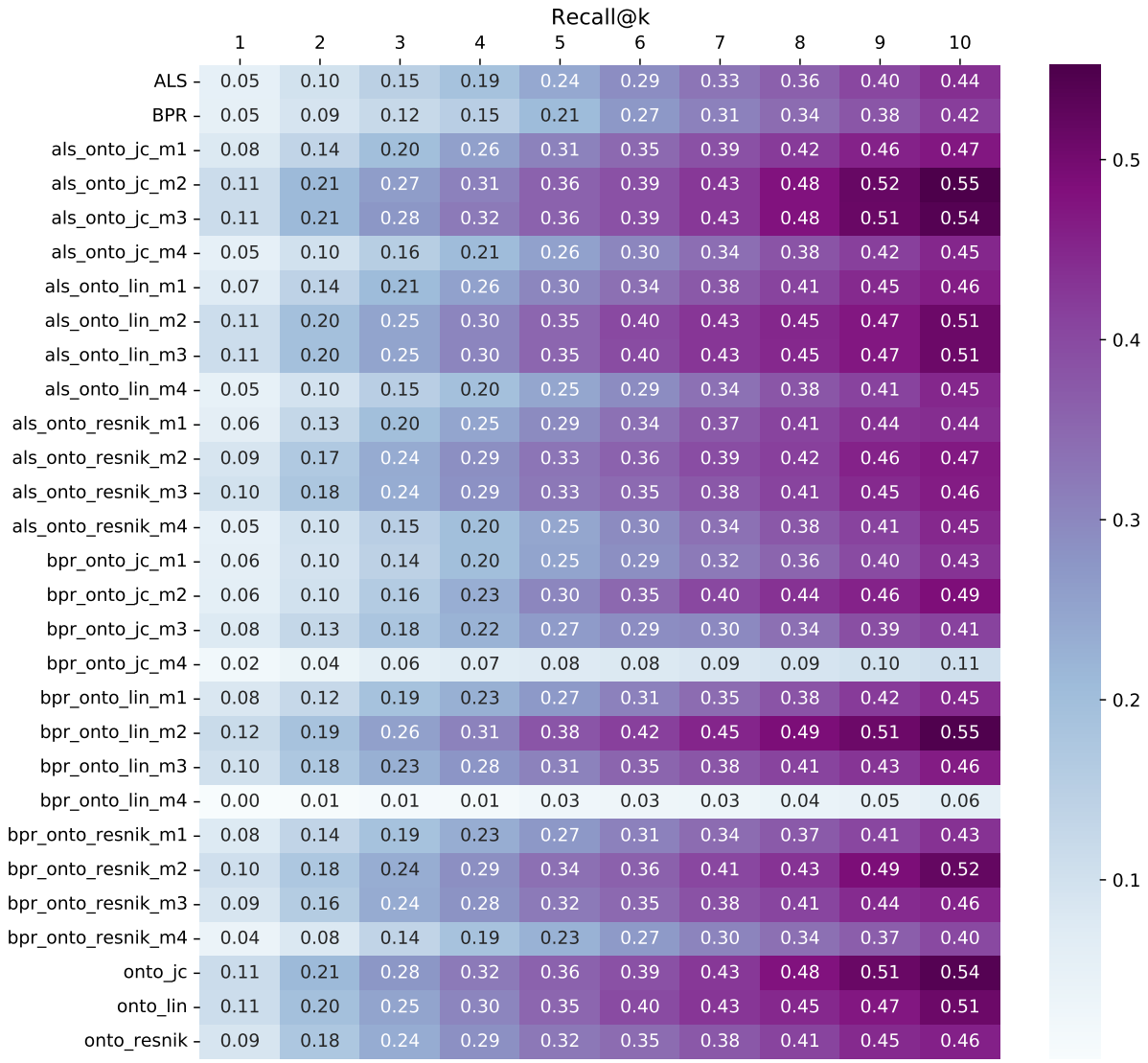


Figure A.6: A list of top@10 Recall results from Med4EMC-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

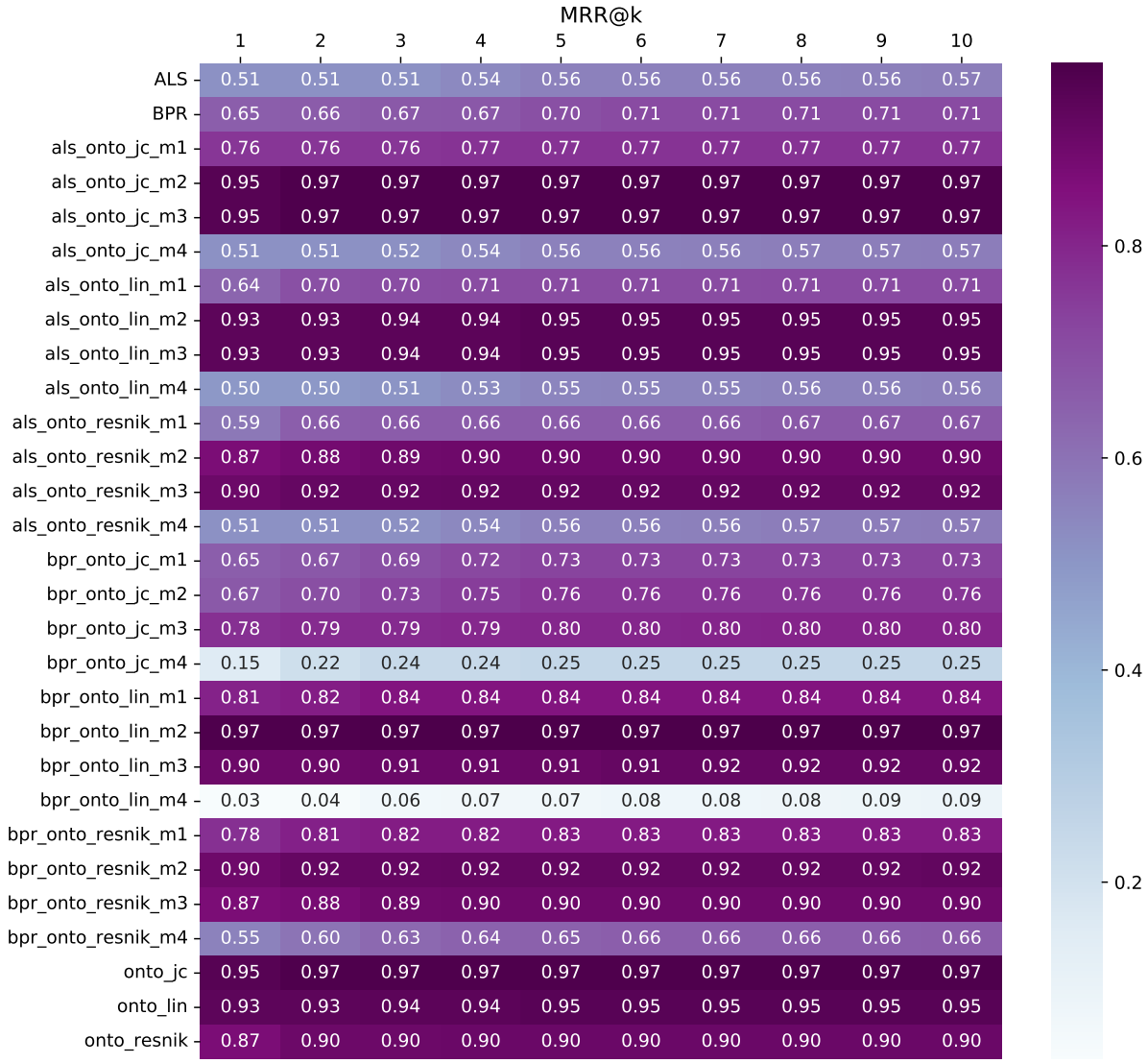


Figure A.7: A list of top@10 MRR results from Med4EMC-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

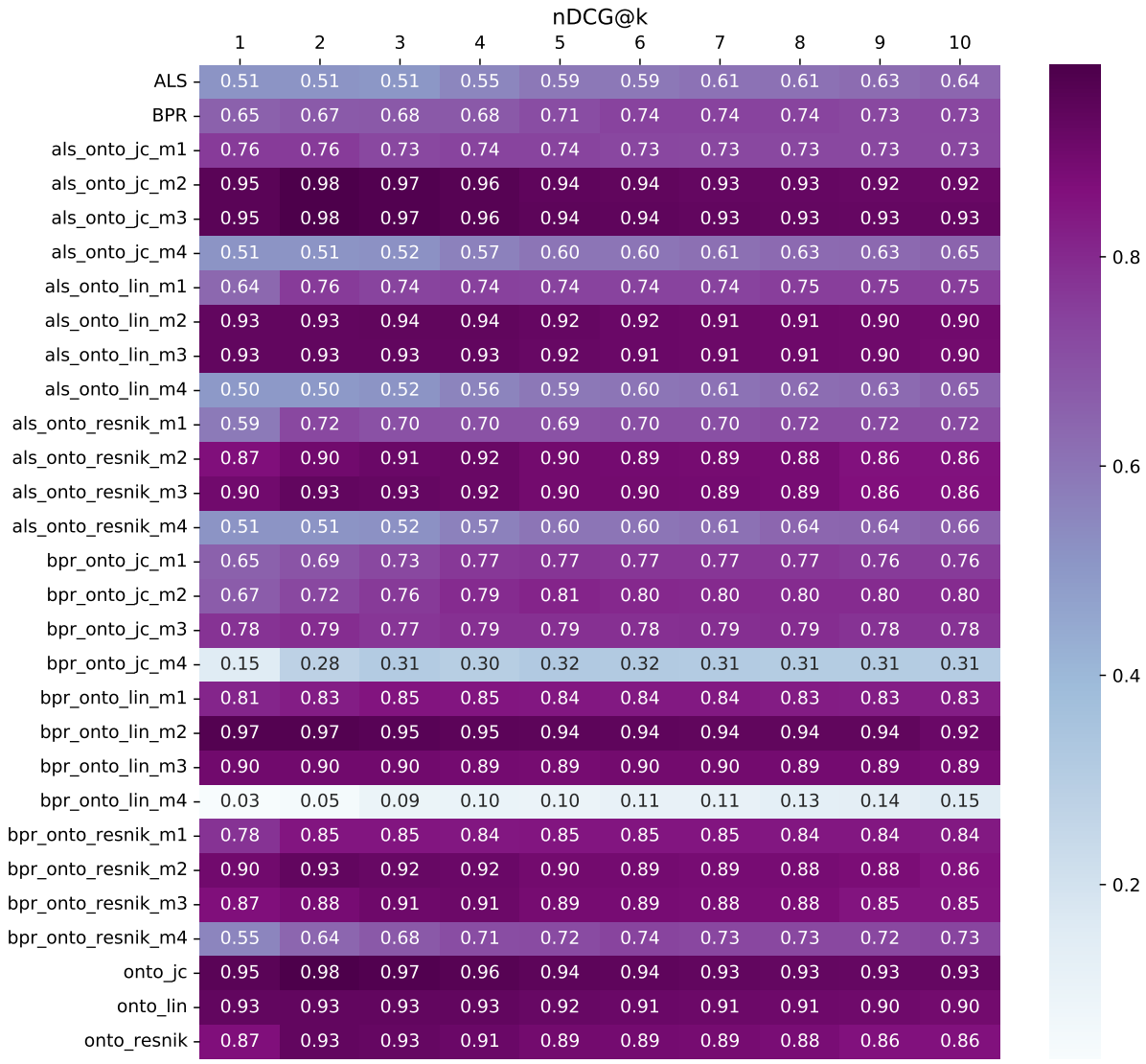


Figure A.8: A list of top@10 nDCG results from Med4EMC-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

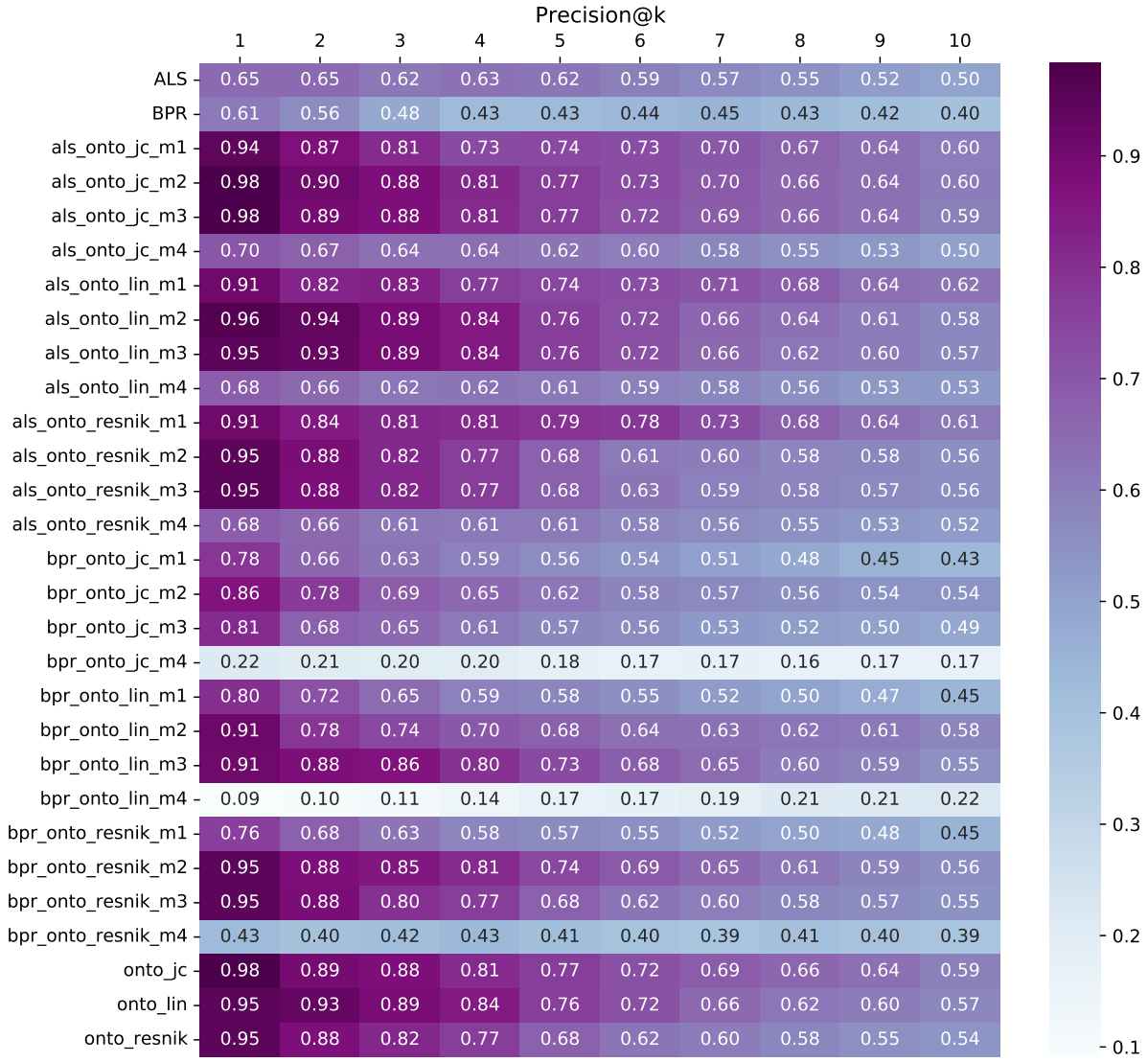


Figure A.9: A list of top@10 Precision results from Med4EMC-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

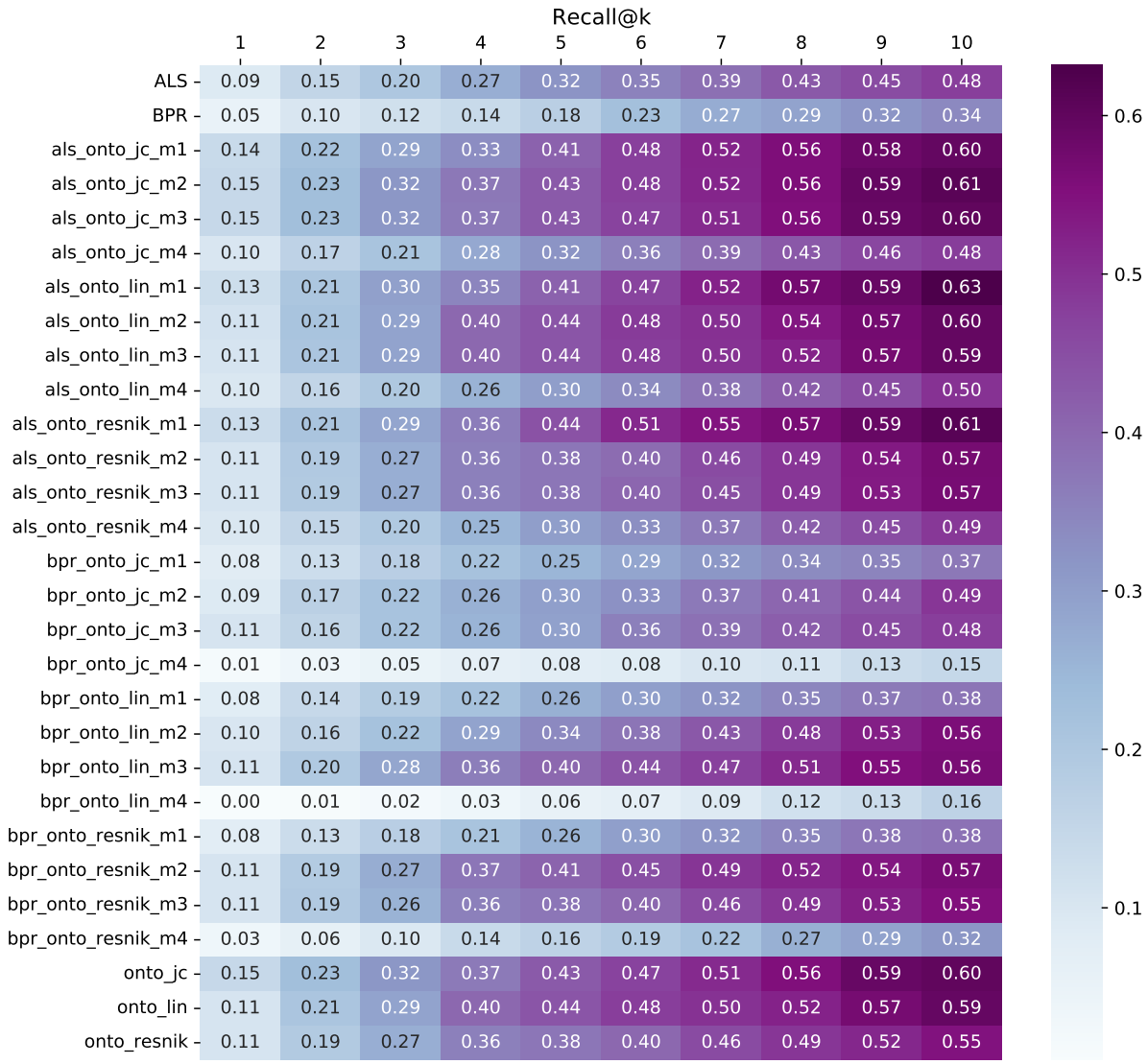


Figure A.10: A list of top@10 Recall results from Med4EMC-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

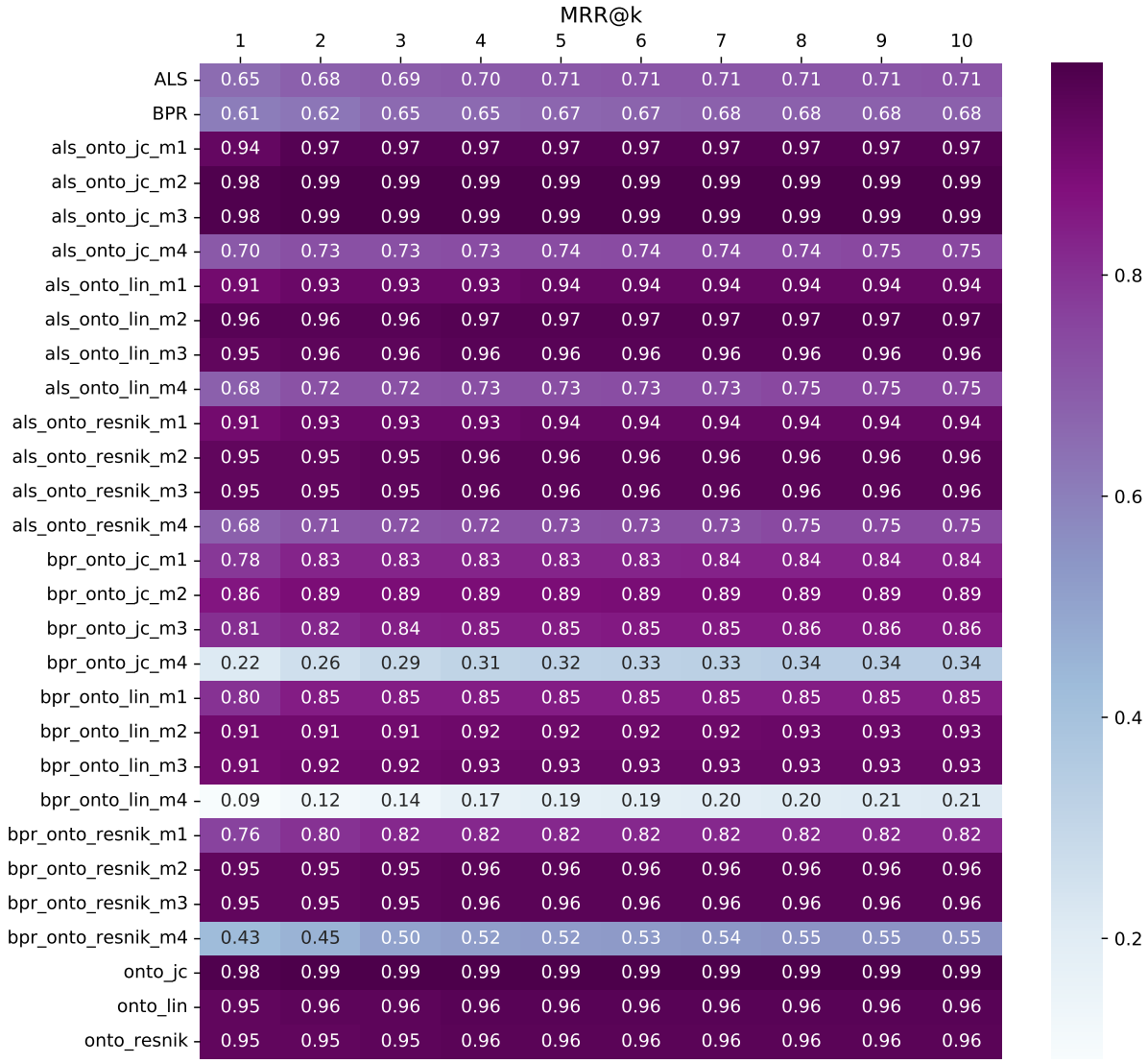


Figure A.11: A list of top@10 MRR results from Med4EMC-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

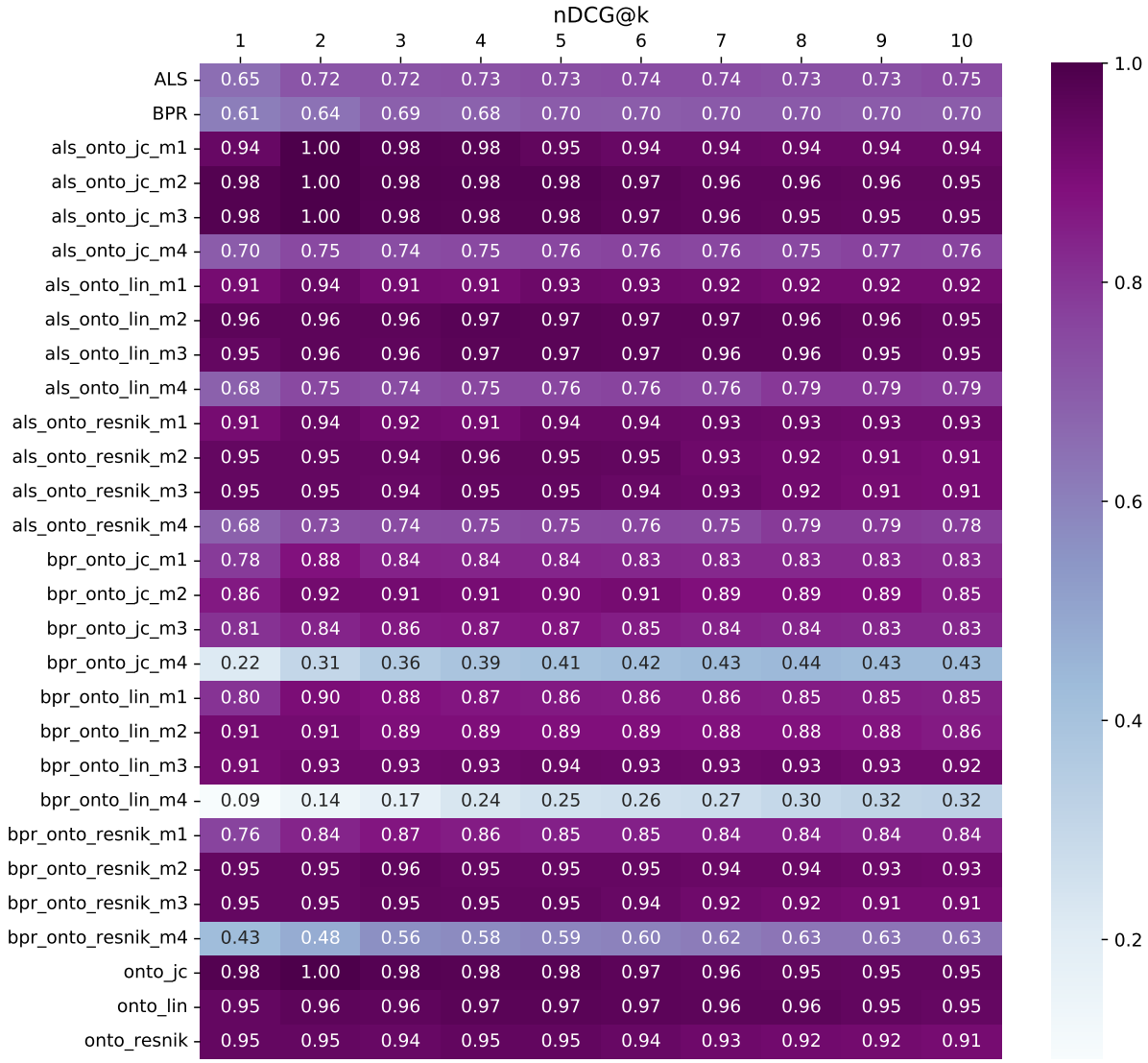


Figure A.12: A list of top@10 nDCG results from Med4EMC-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

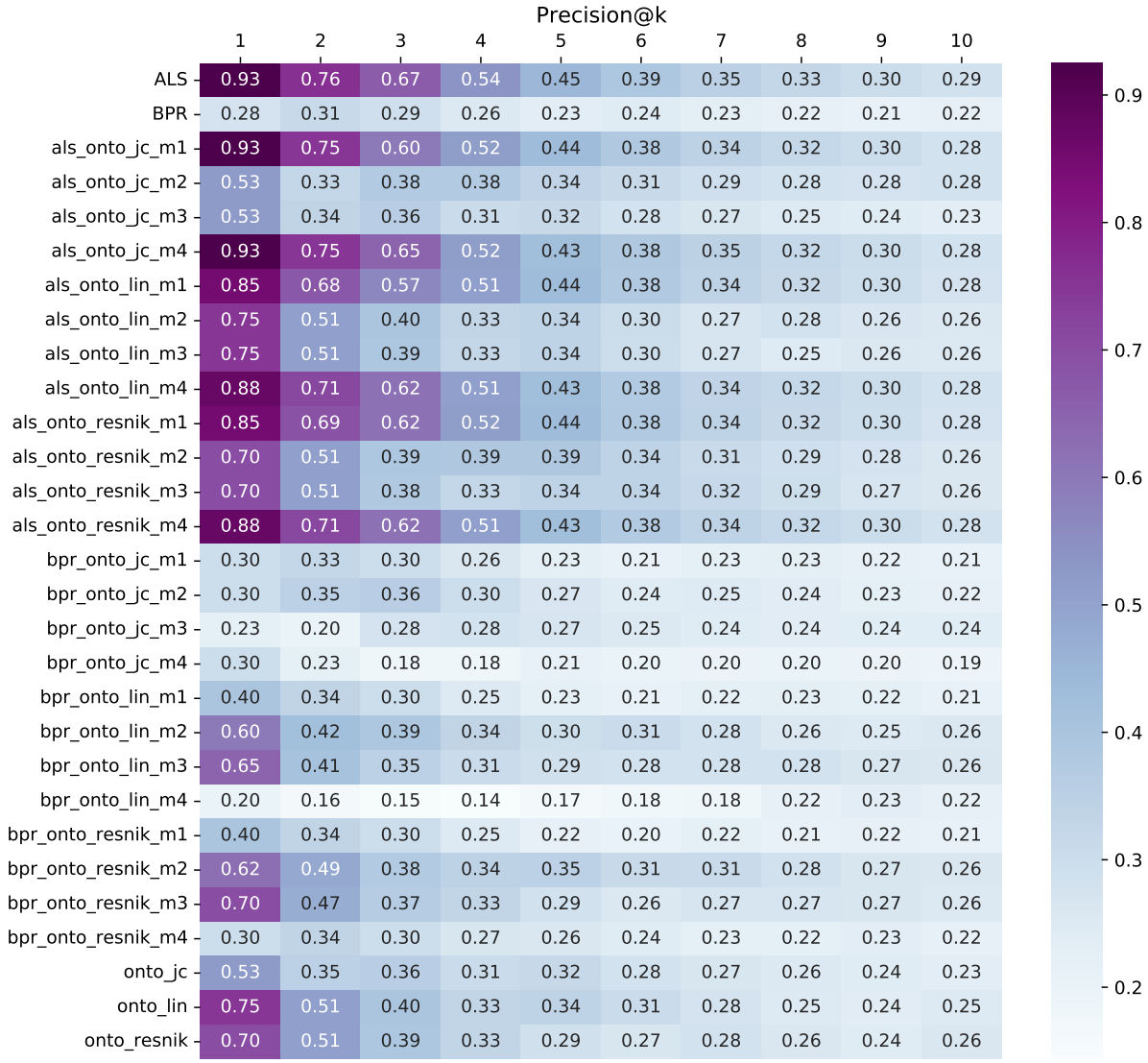


Figure A.13: A list of top@10 Precision results from Med4EMC-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

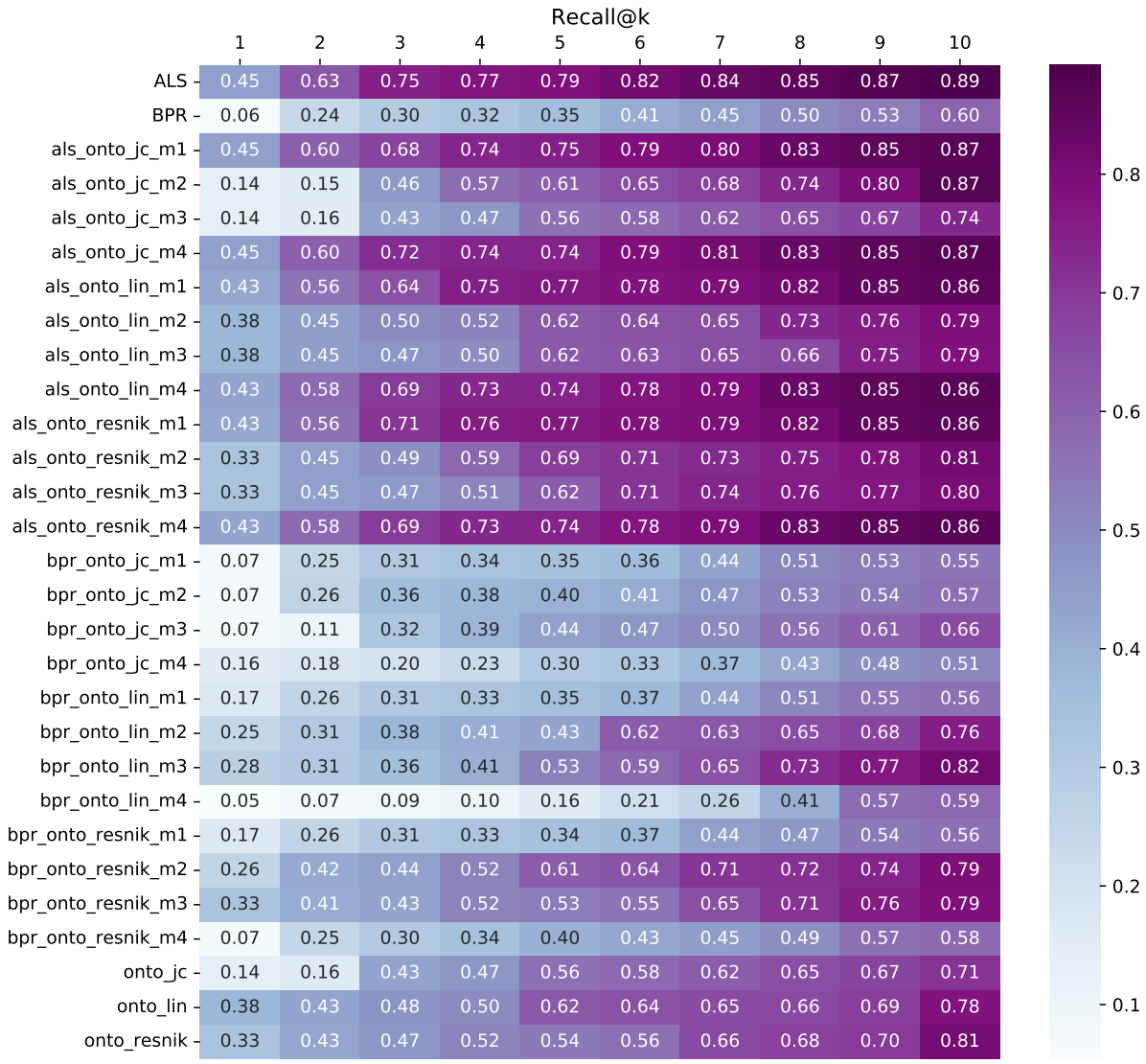


Figure A.14: A list of top@10 Recall results from Med4EMC-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

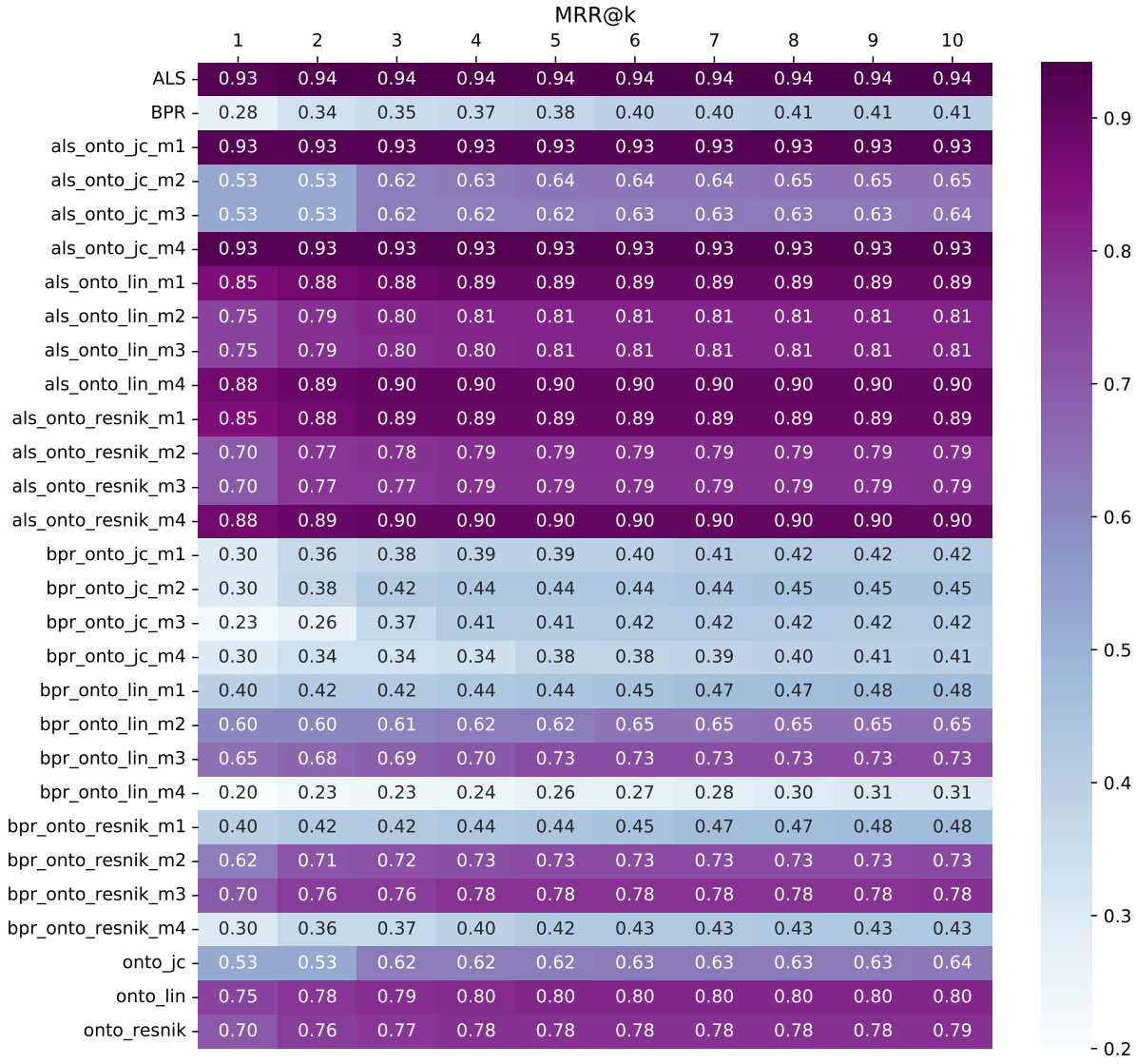


Figure A.15: A list of top@10 MRR results from Med4EMC-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

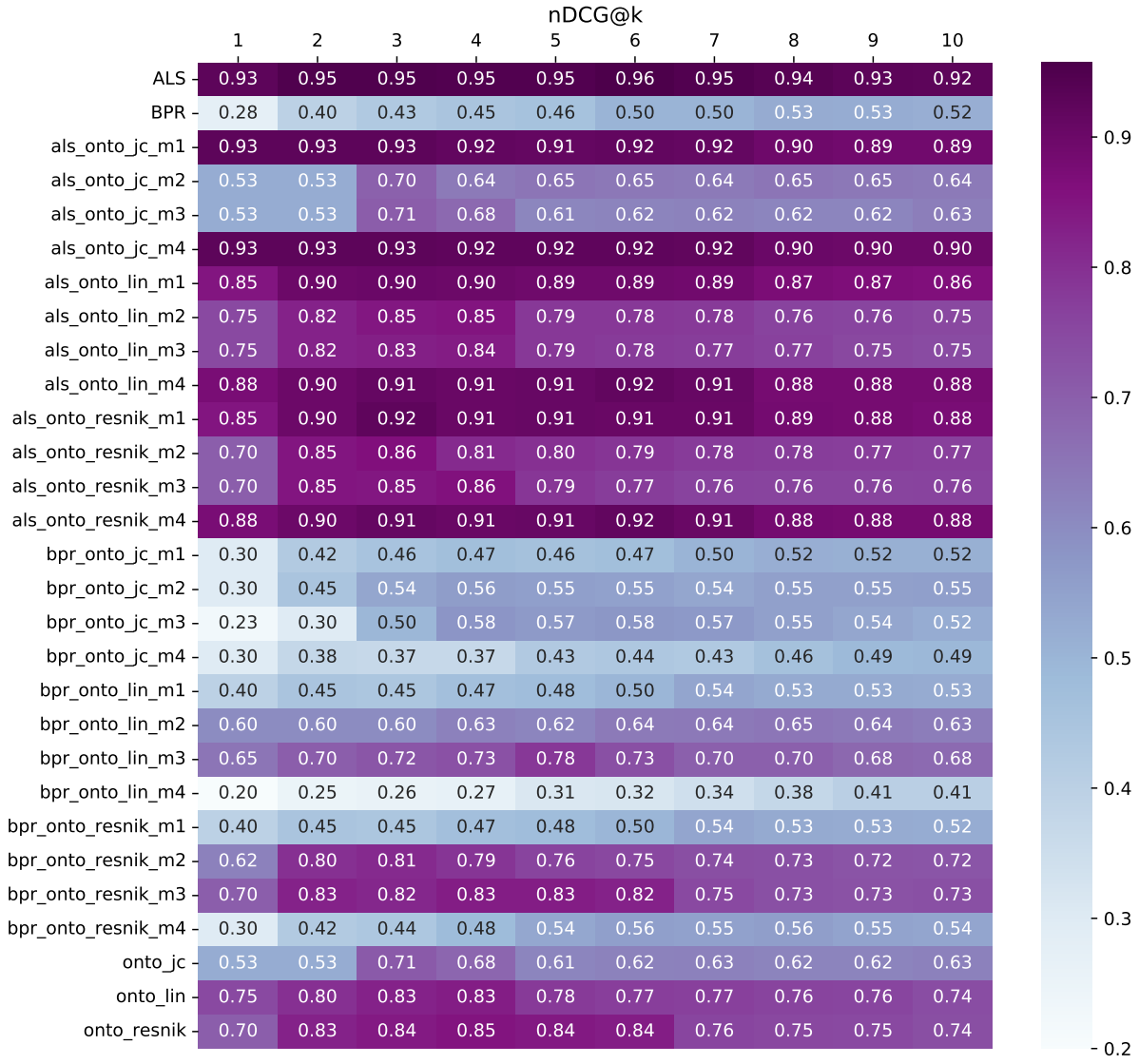


Figure A.16: A list of top@10 nDCG results from Med4EMC-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

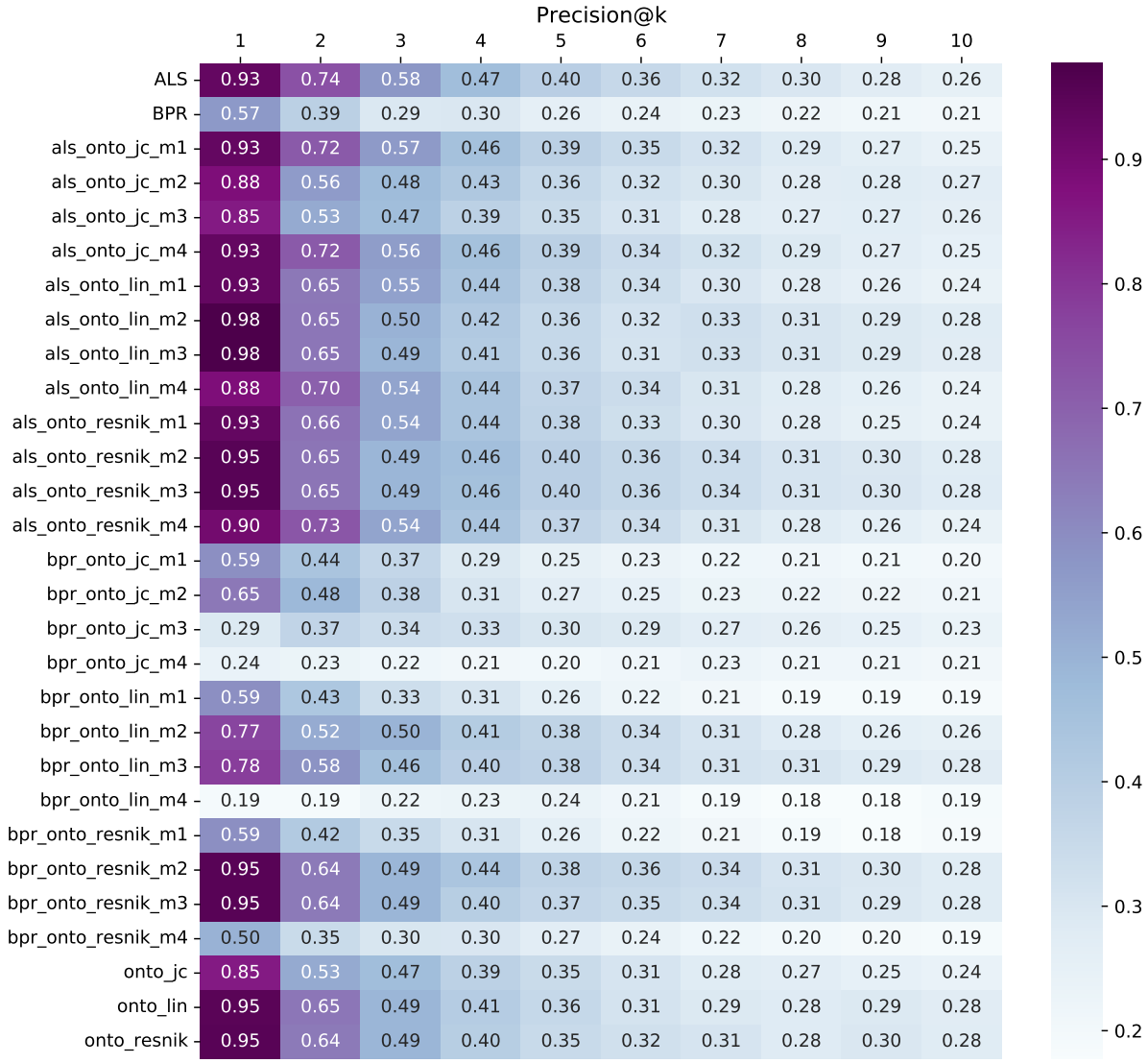


Figure A.17: A list of top@10 Precision results from Med4EMC-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items in calculating the ONTO scores.

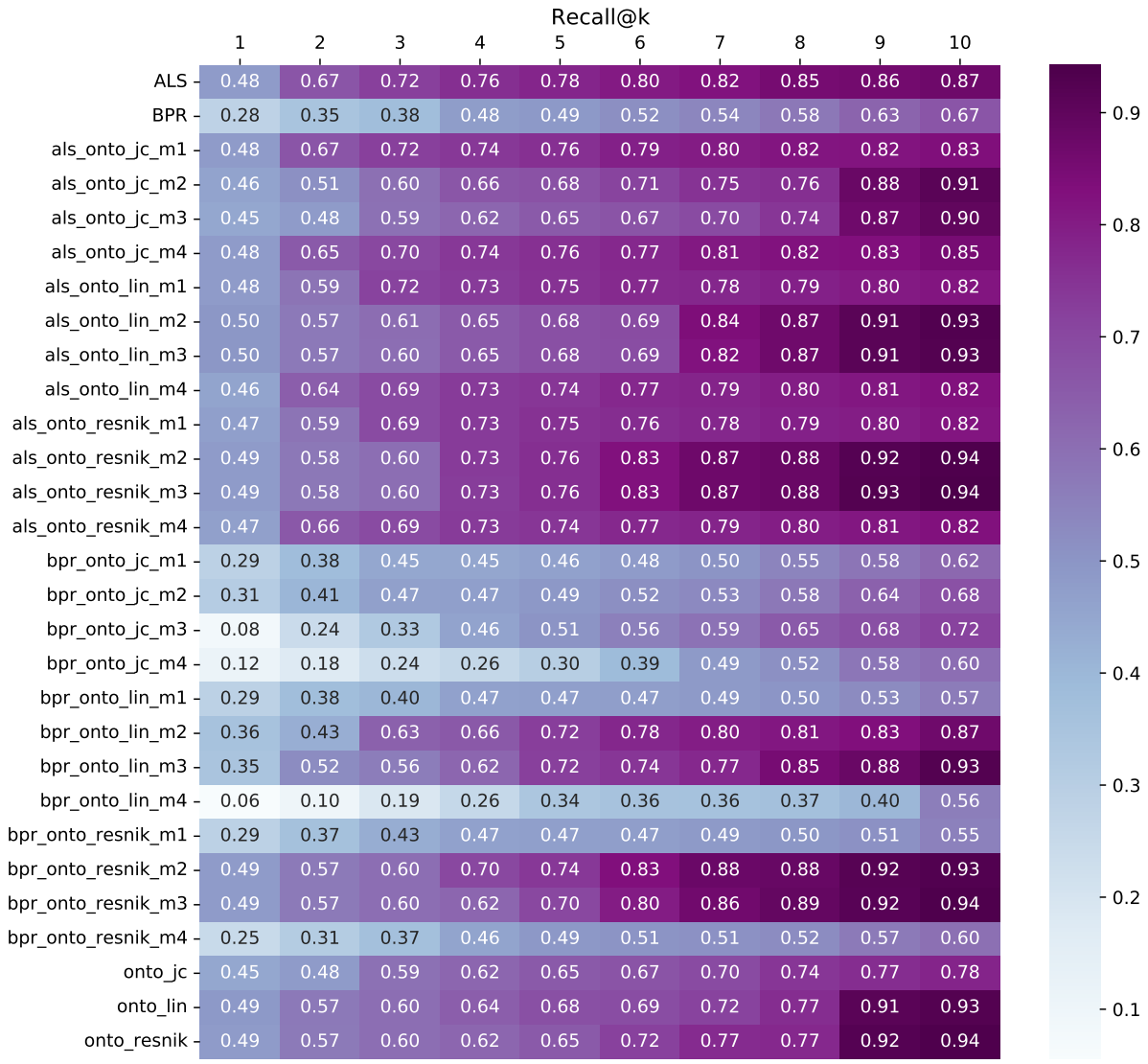


Figure A.18: A list of top@10 Recall results from Med4EMC-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

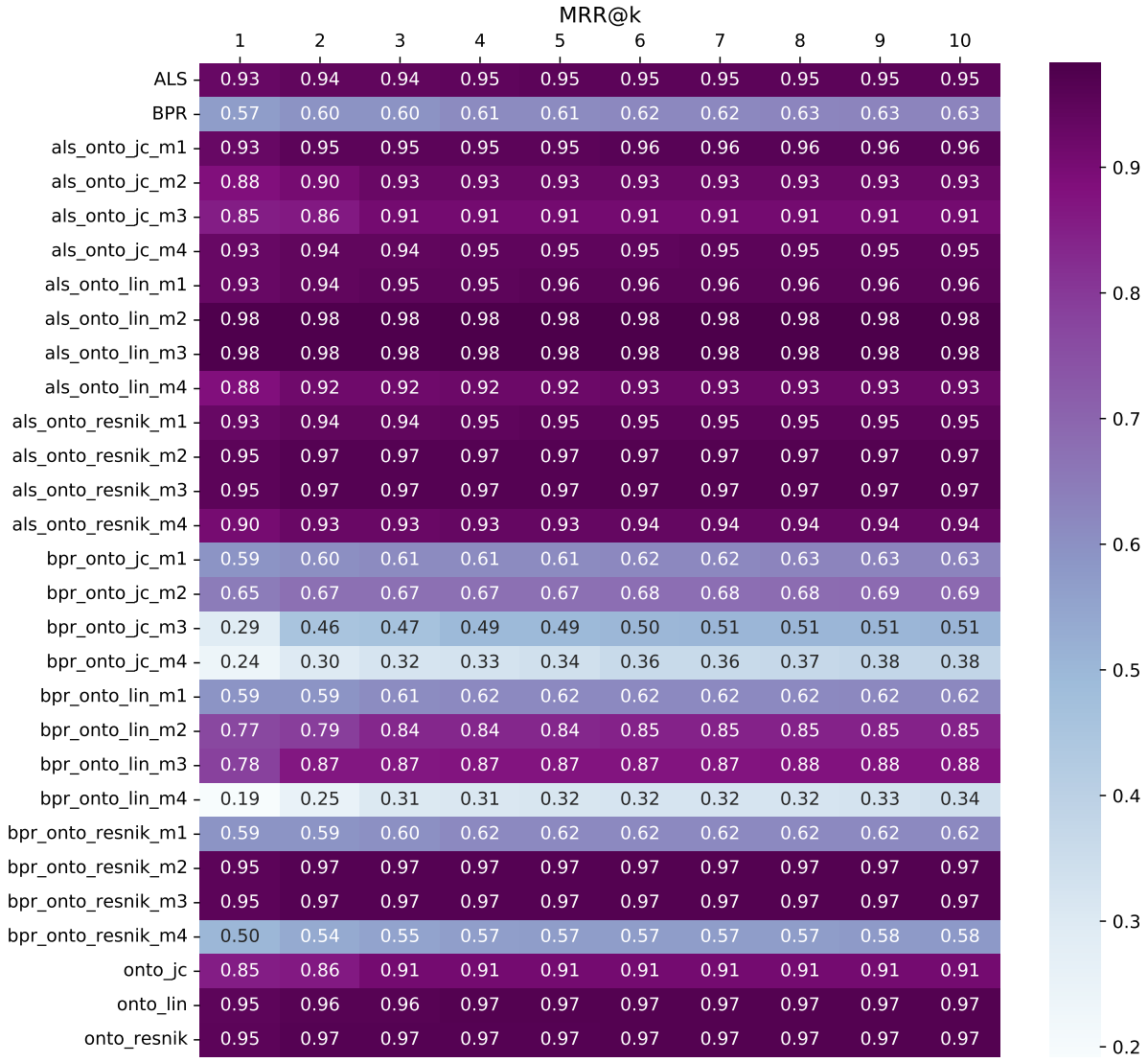


Figure A.19: A list of top@10 MRR results from Med4EMC-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

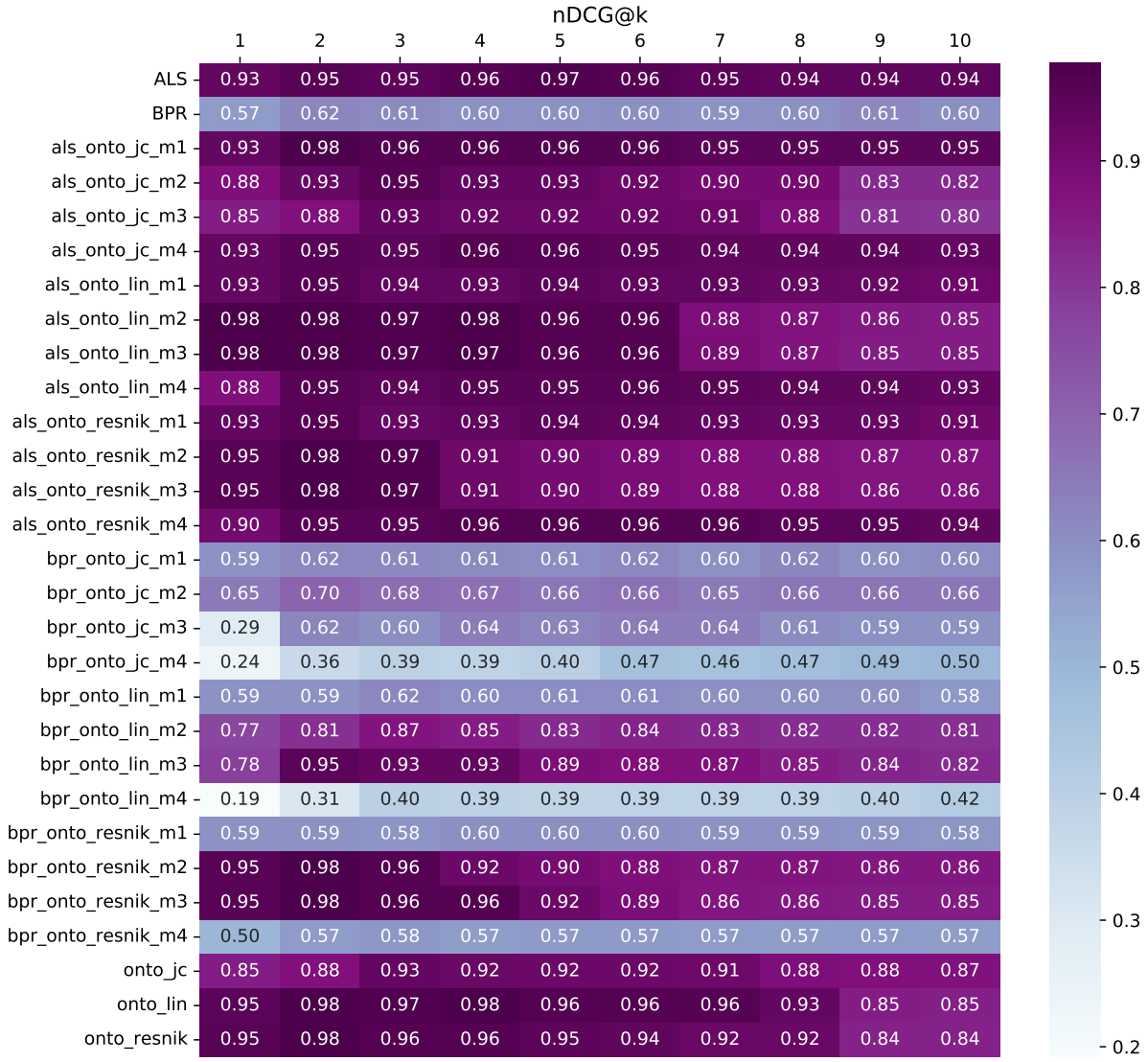


Figure A.20: A list of top@10 nDCG results from Med4EMC-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

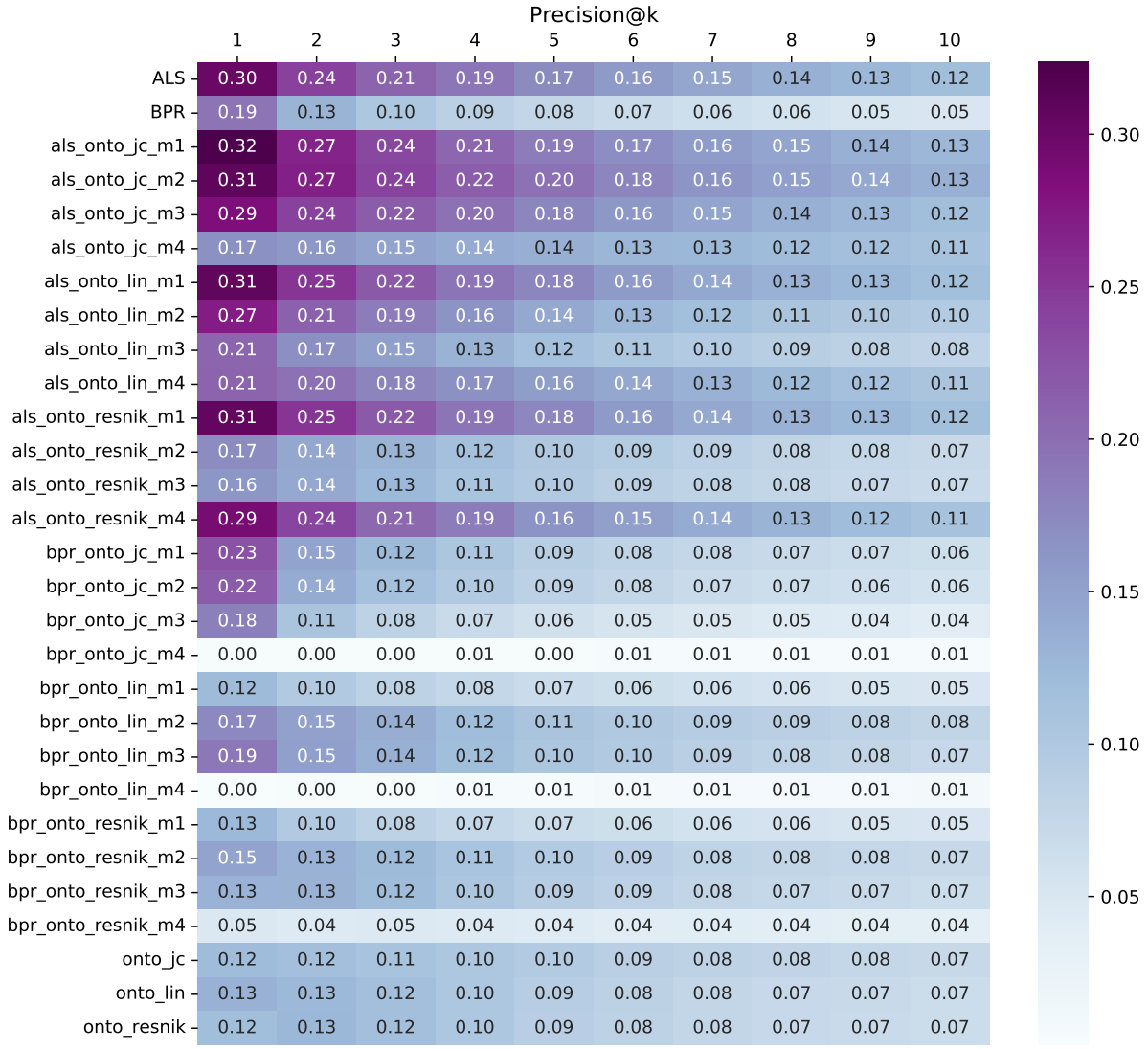


Figure A.21: A list of top@10 Precision results from Med4DB, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

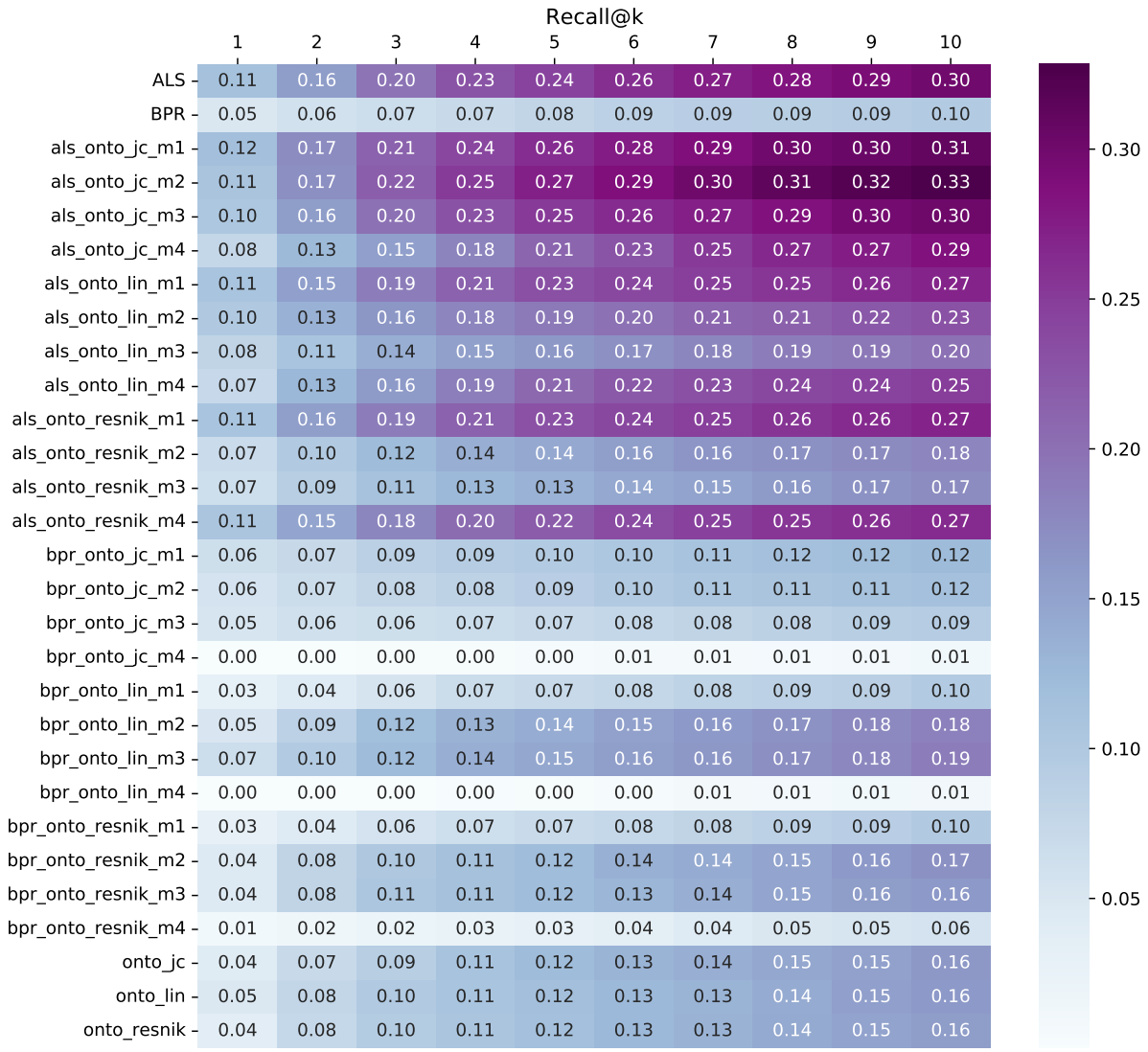


Figure A.22: A list of top@10 Recall results from Med4DB, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

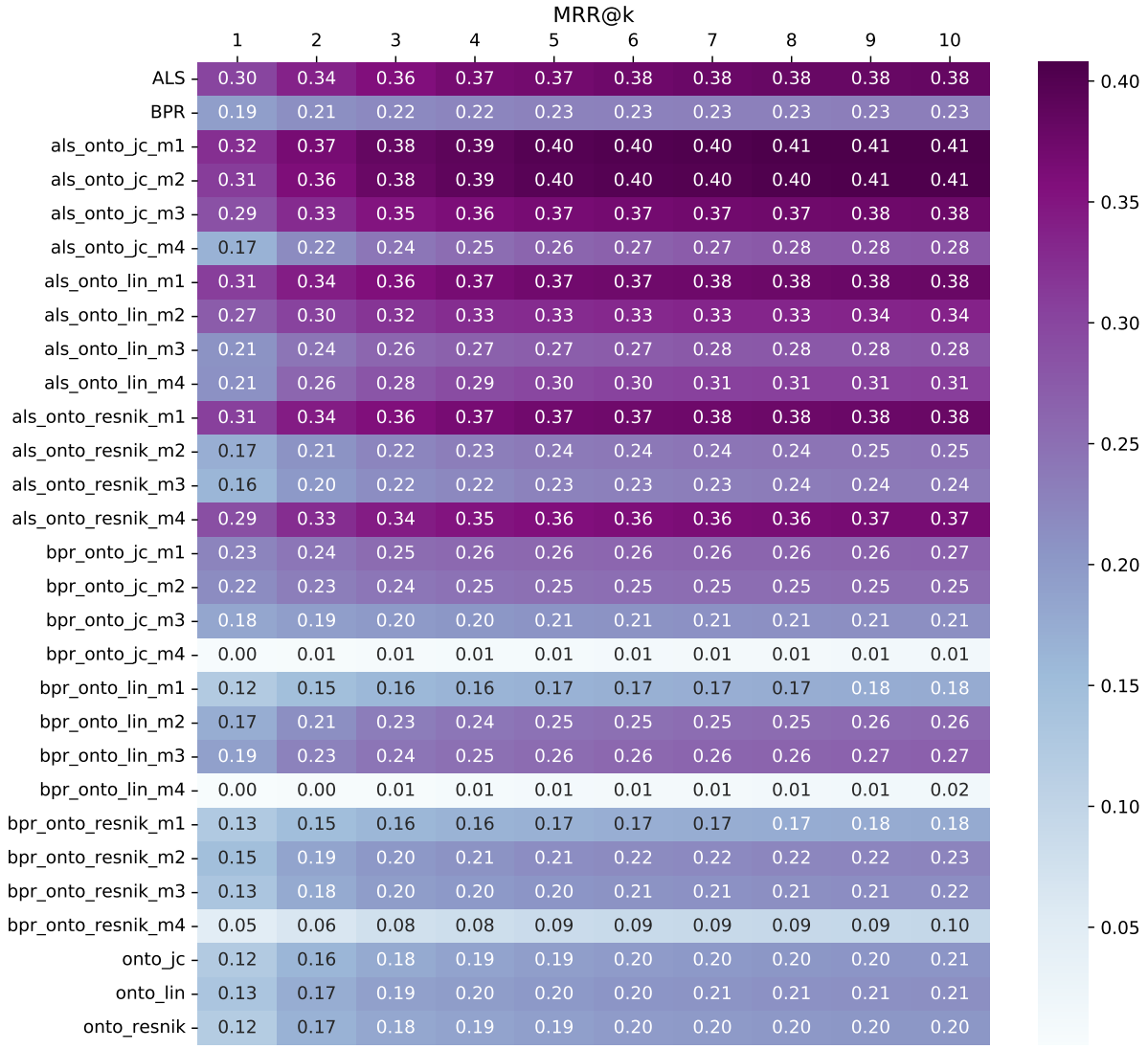


Figure A.23: A list of top@10 MRR results from Med4DB, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

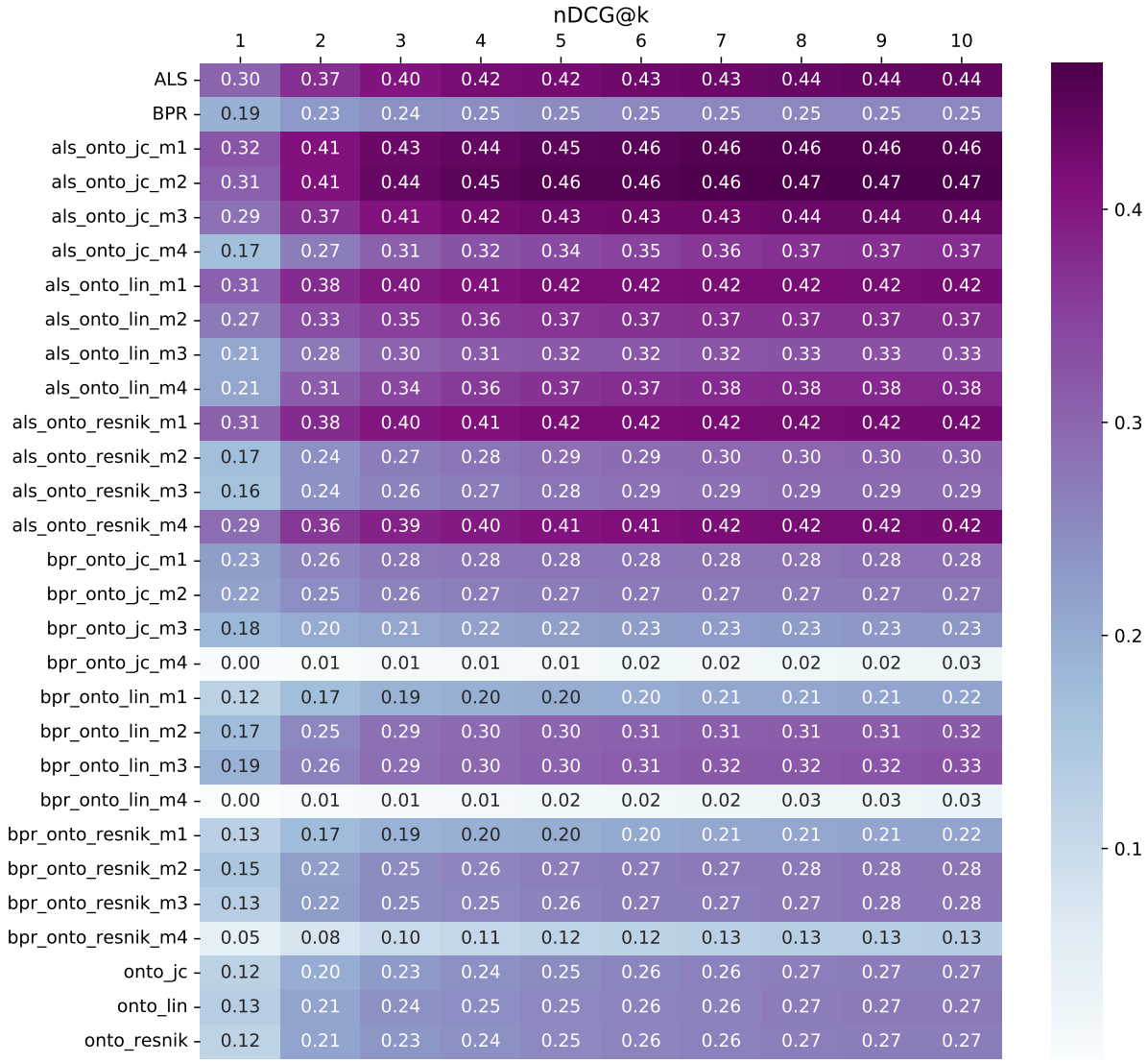


Figure A.24: A list of top@10 nDCG results from Med4DB, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

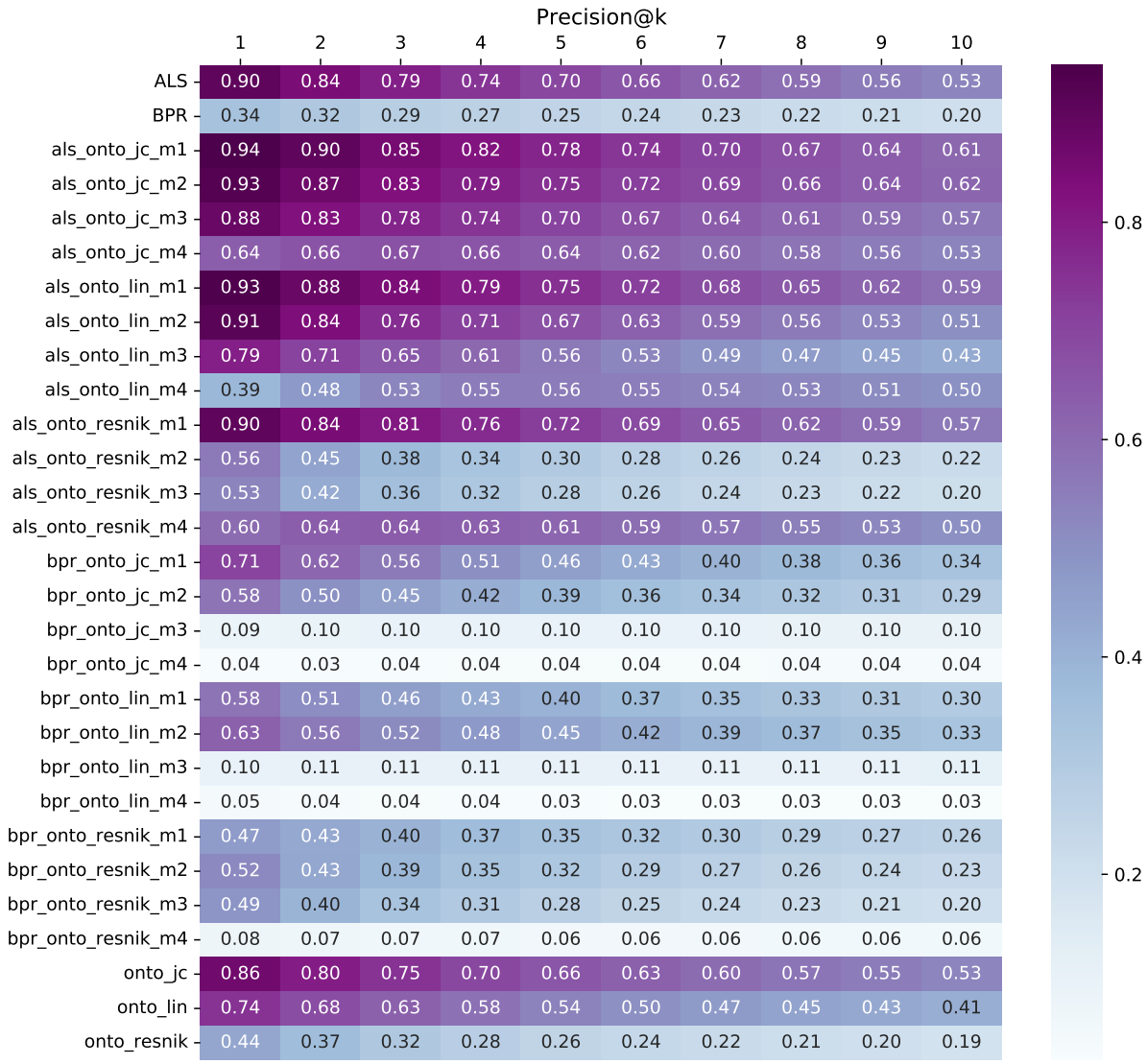


Figure A.25: A list of top@10 Precision results from Med4DB-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

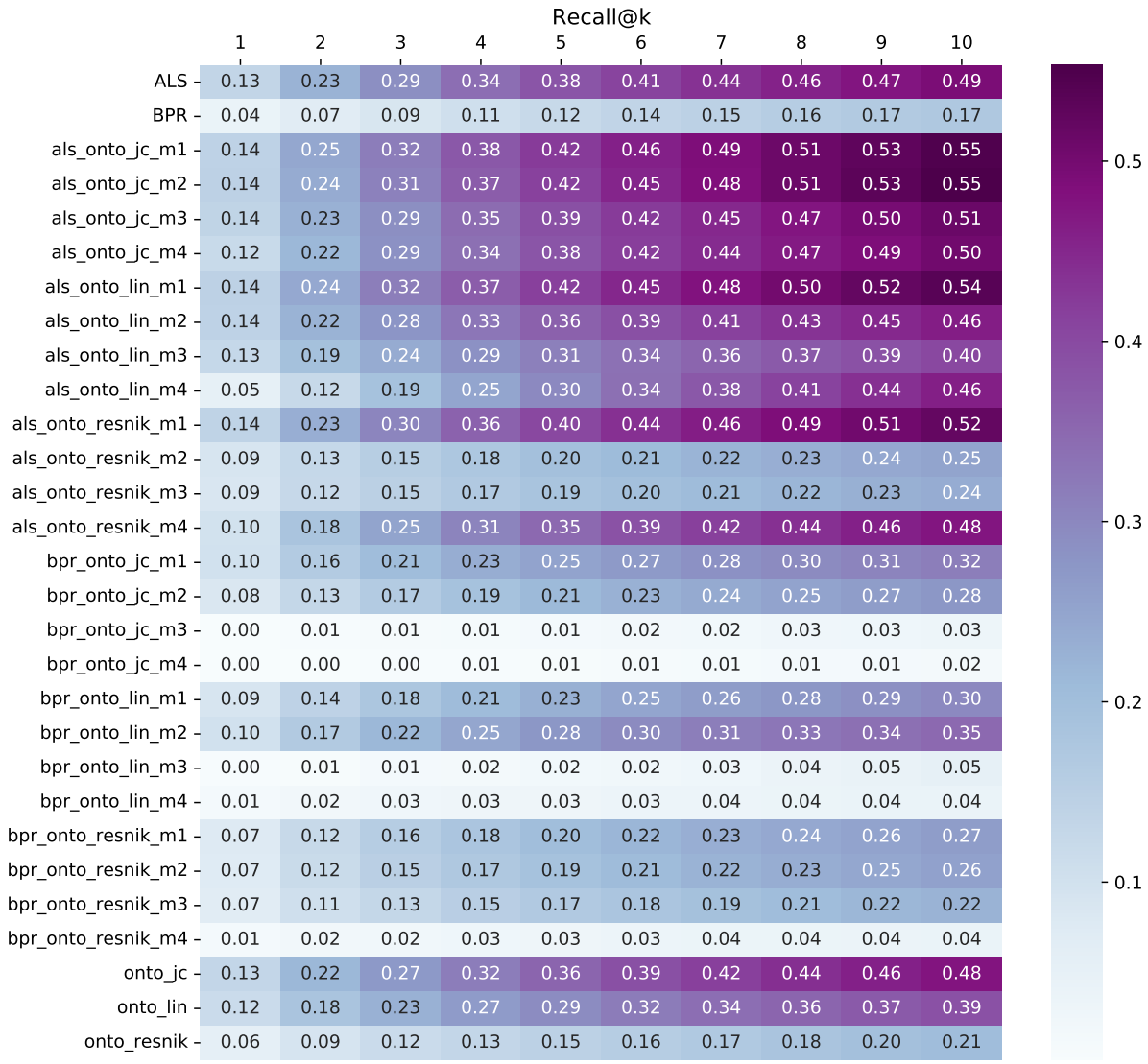


Figure A.26: A list of top@10 Recall results from Med4DB-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

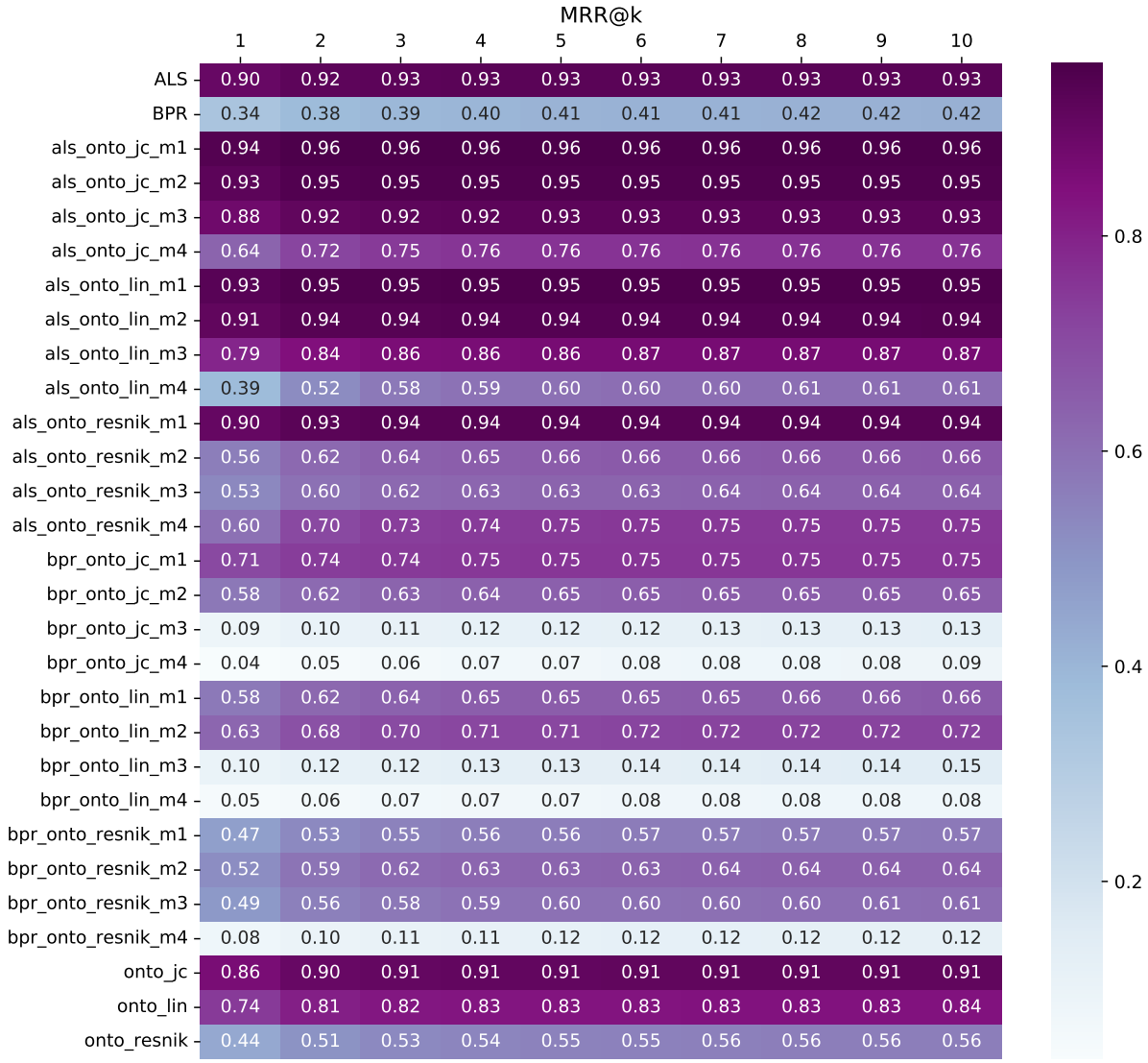


Figure A.27: A list of top@10 MRR results from Med4DB-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

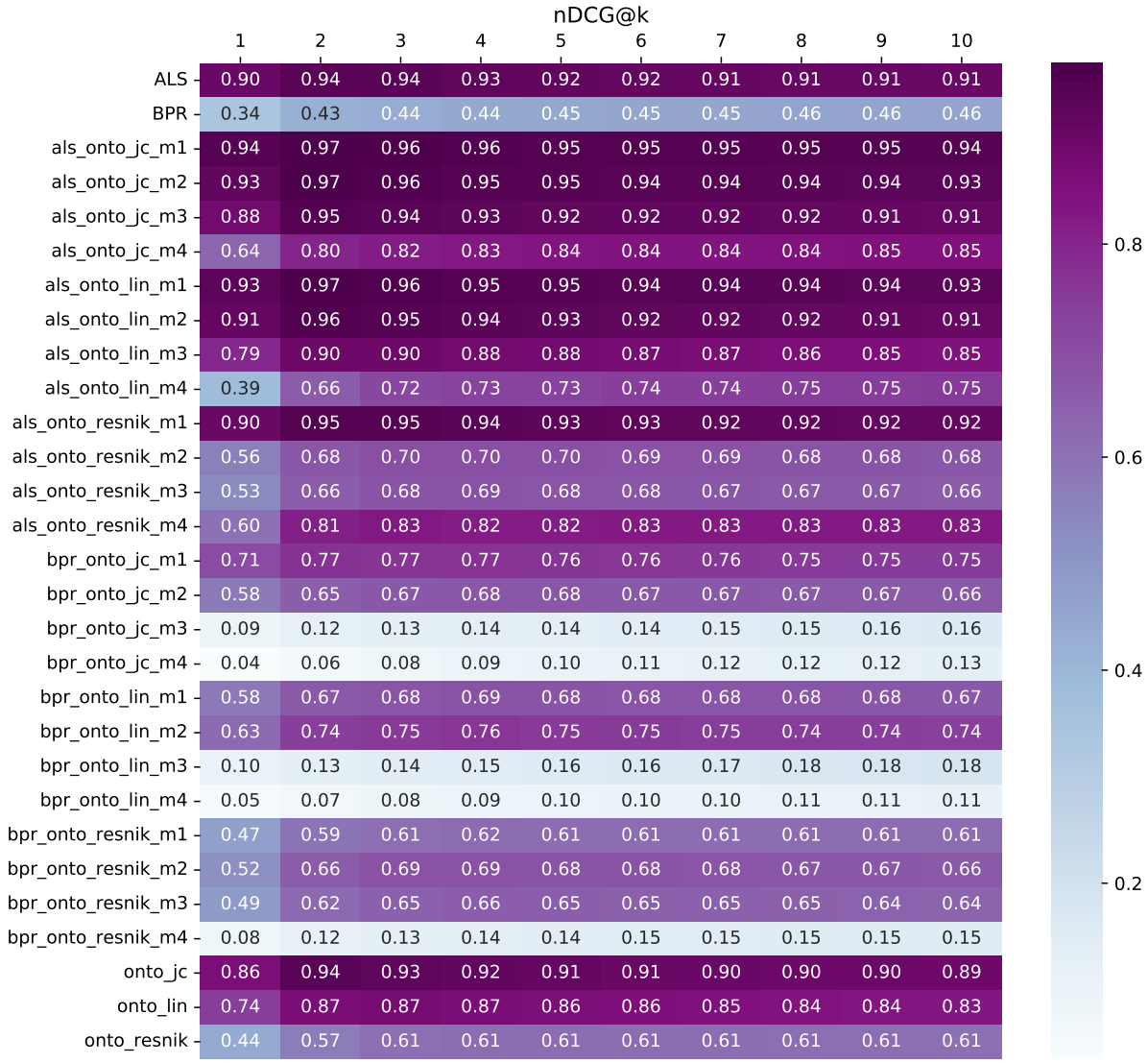


Figure A.28: A list of top@10 nDCG results from Med4DB-KG-Resnik, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

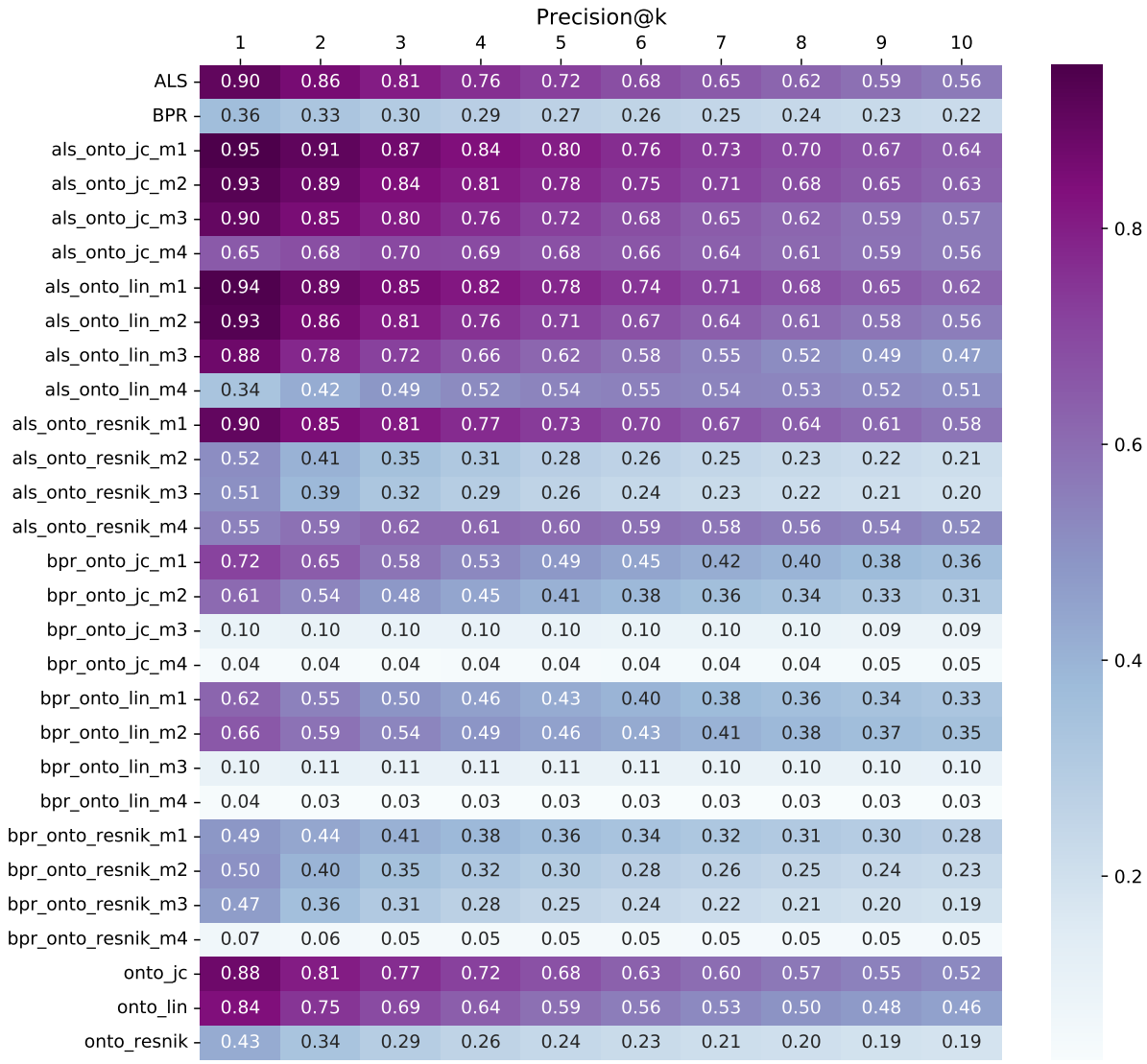


Figure A.29: A list of top@10 Precision results from Med4DB-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

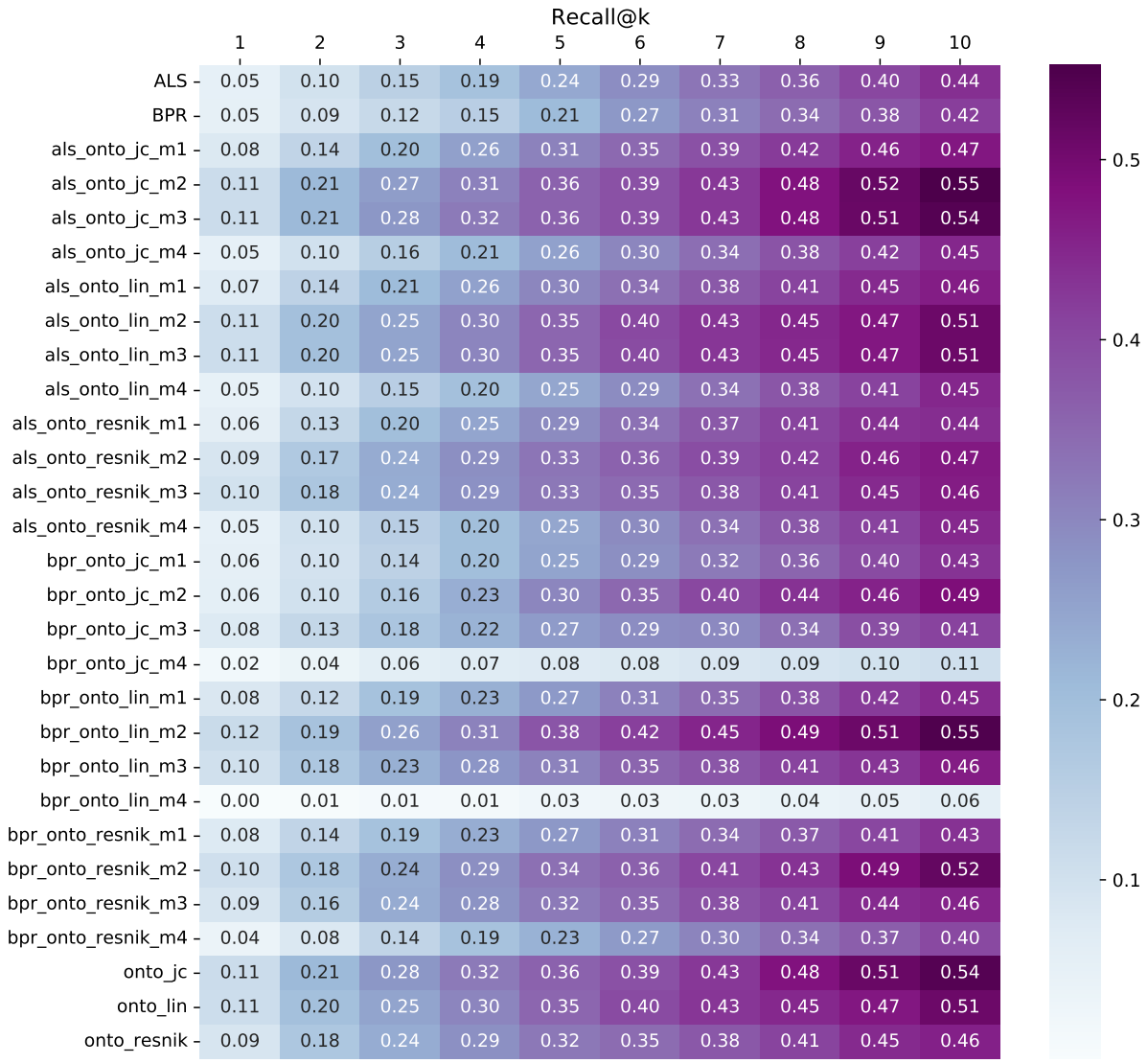


Figure A.30: A list of top@10 Recall results from Med4DB-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

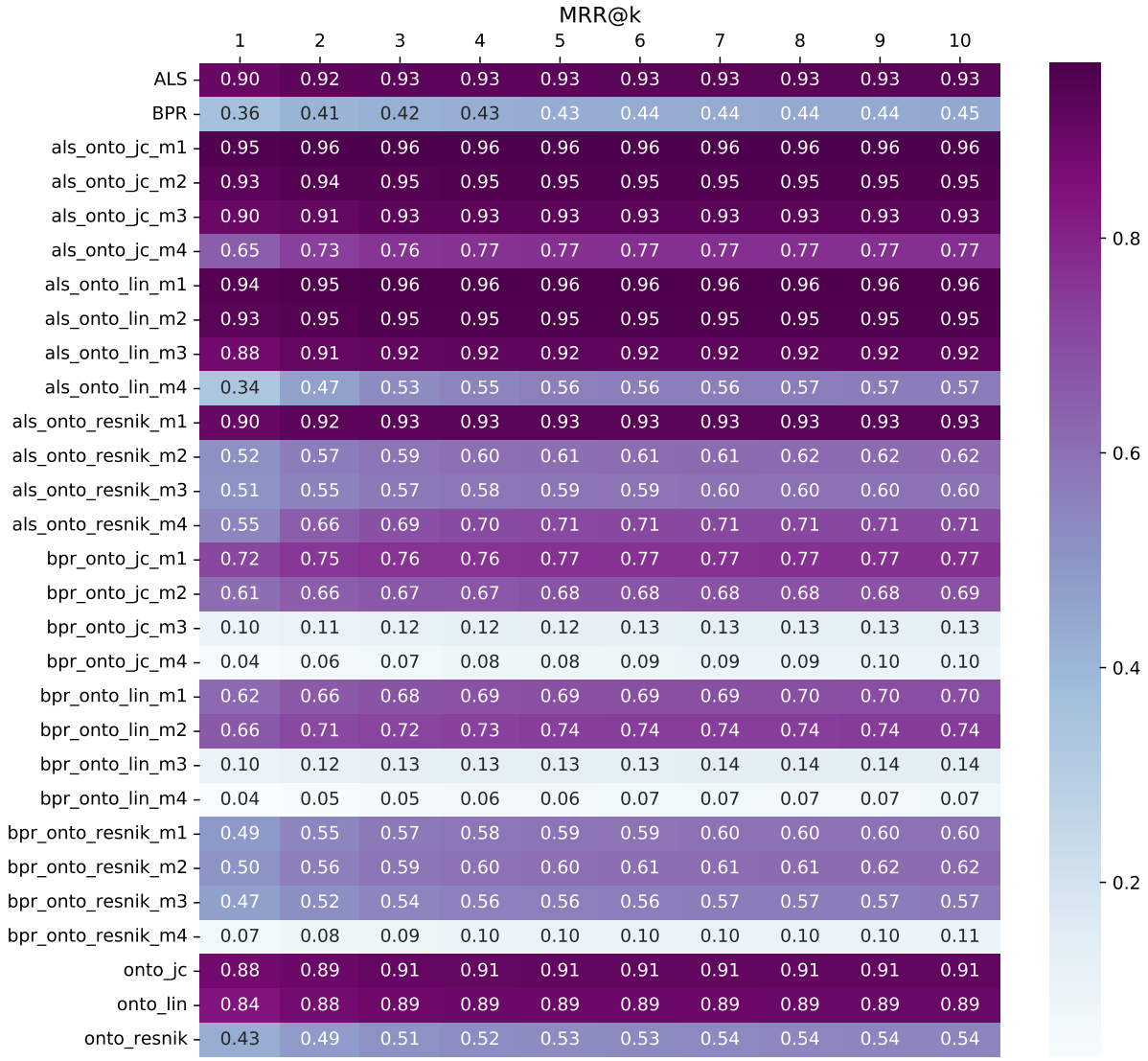


Figure A.31: A list of top@10 MRR results from Med4DB-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

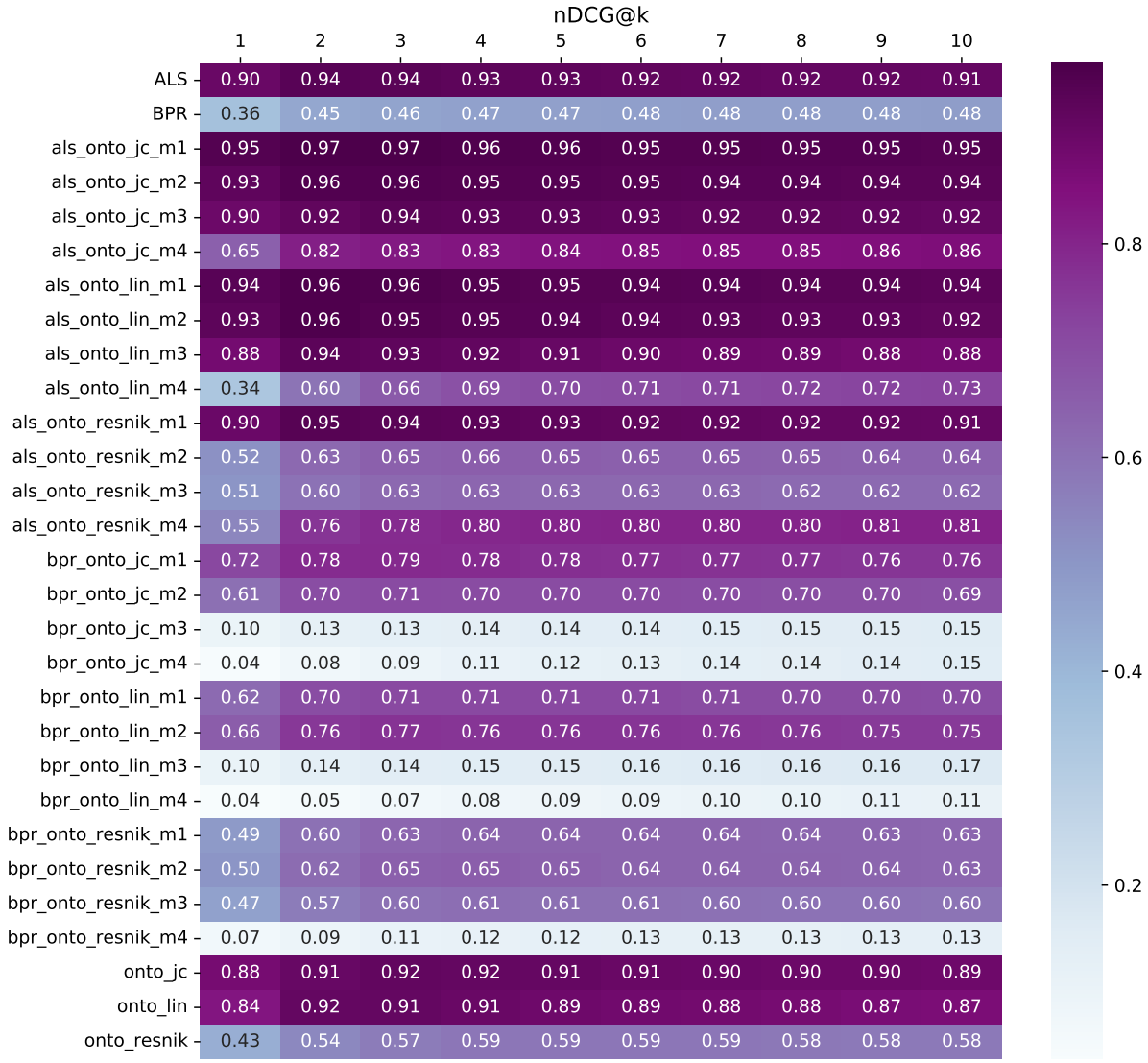


Figure A.32: A list of top@10 nDCG results from Med4DB-KG-JC, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

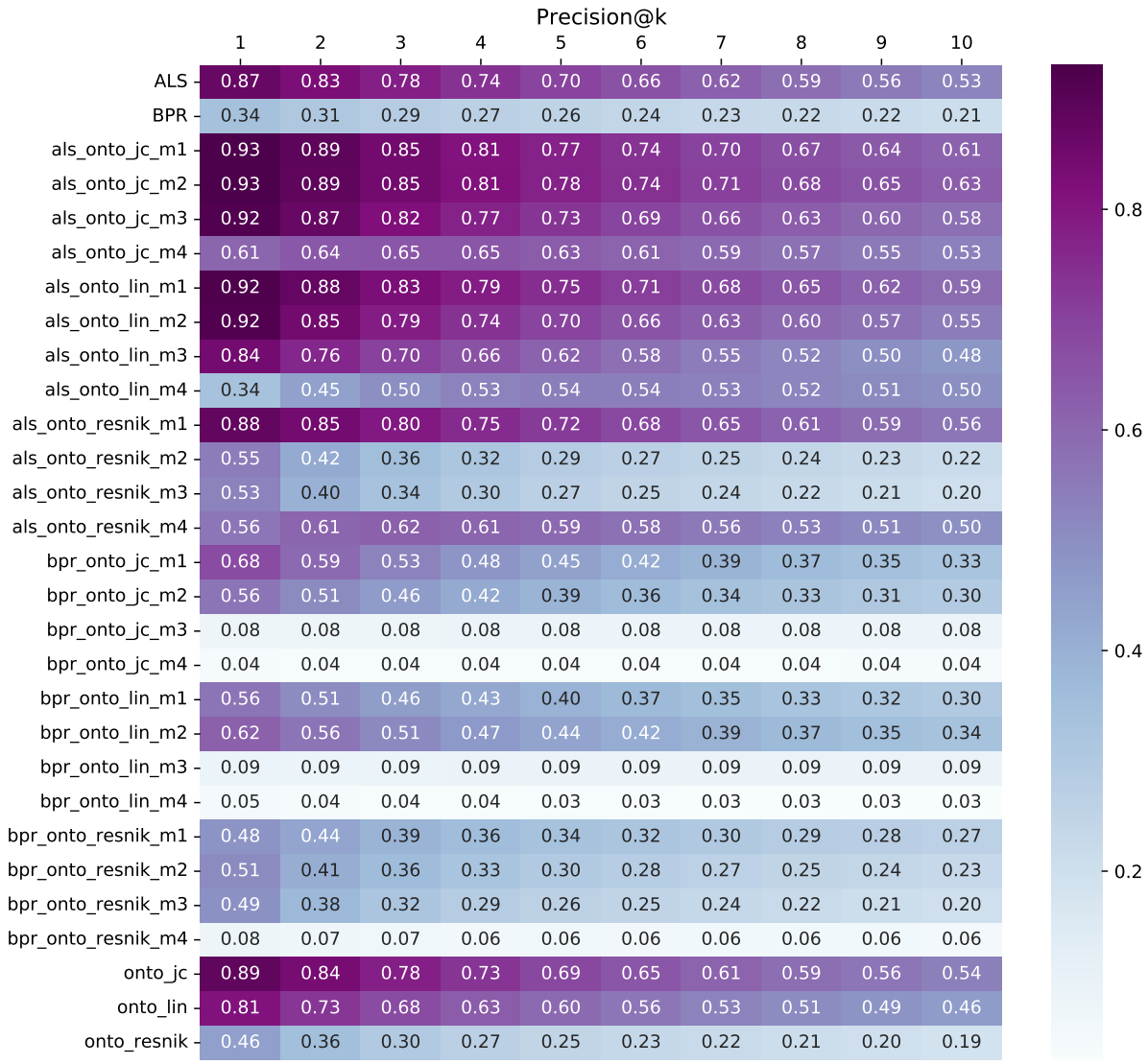


Figure A.33: A list of top@10 Precision results from Med4DB-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

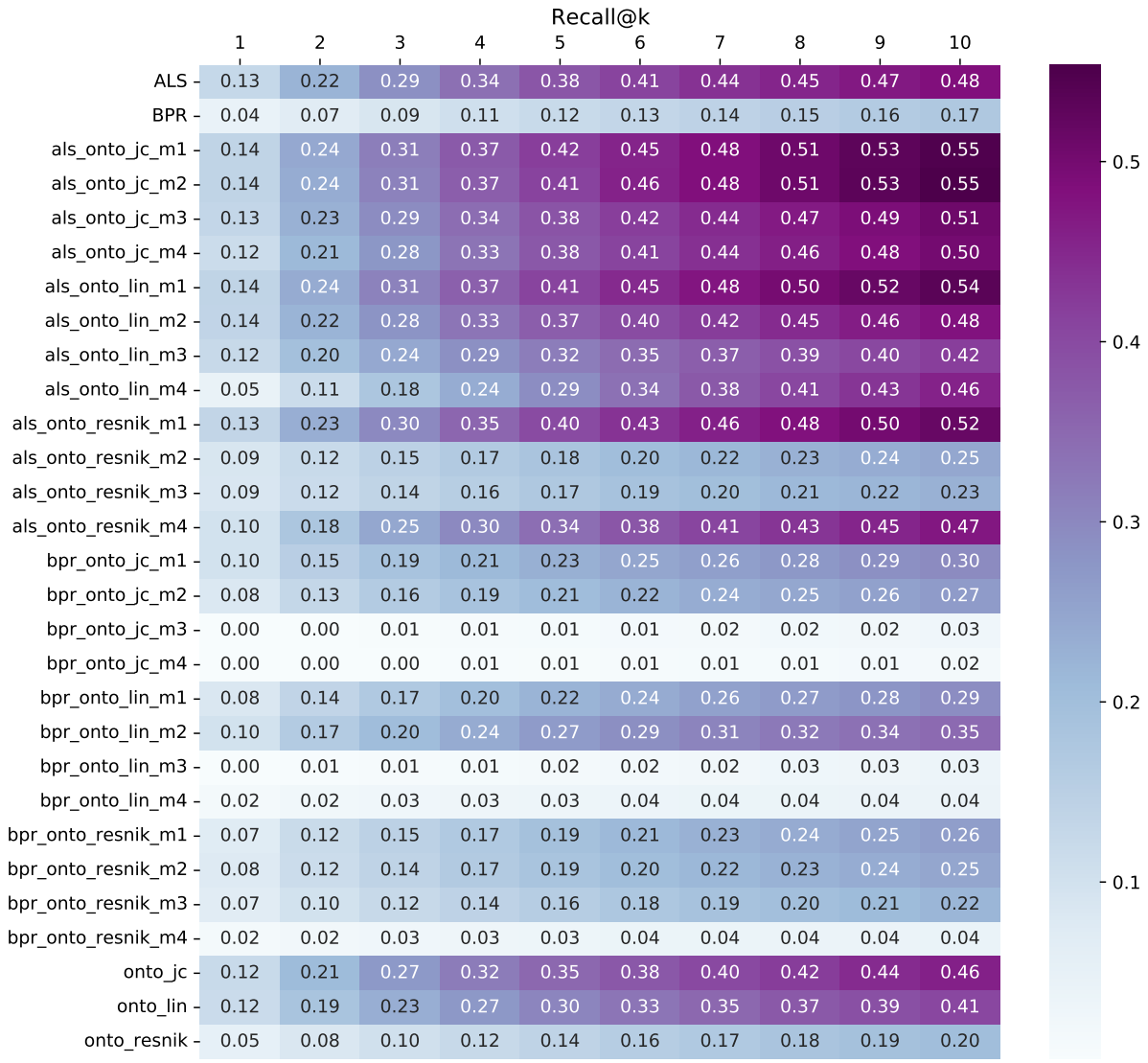


Figure A.34: A list of top@10 Recall results from Med4DB-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

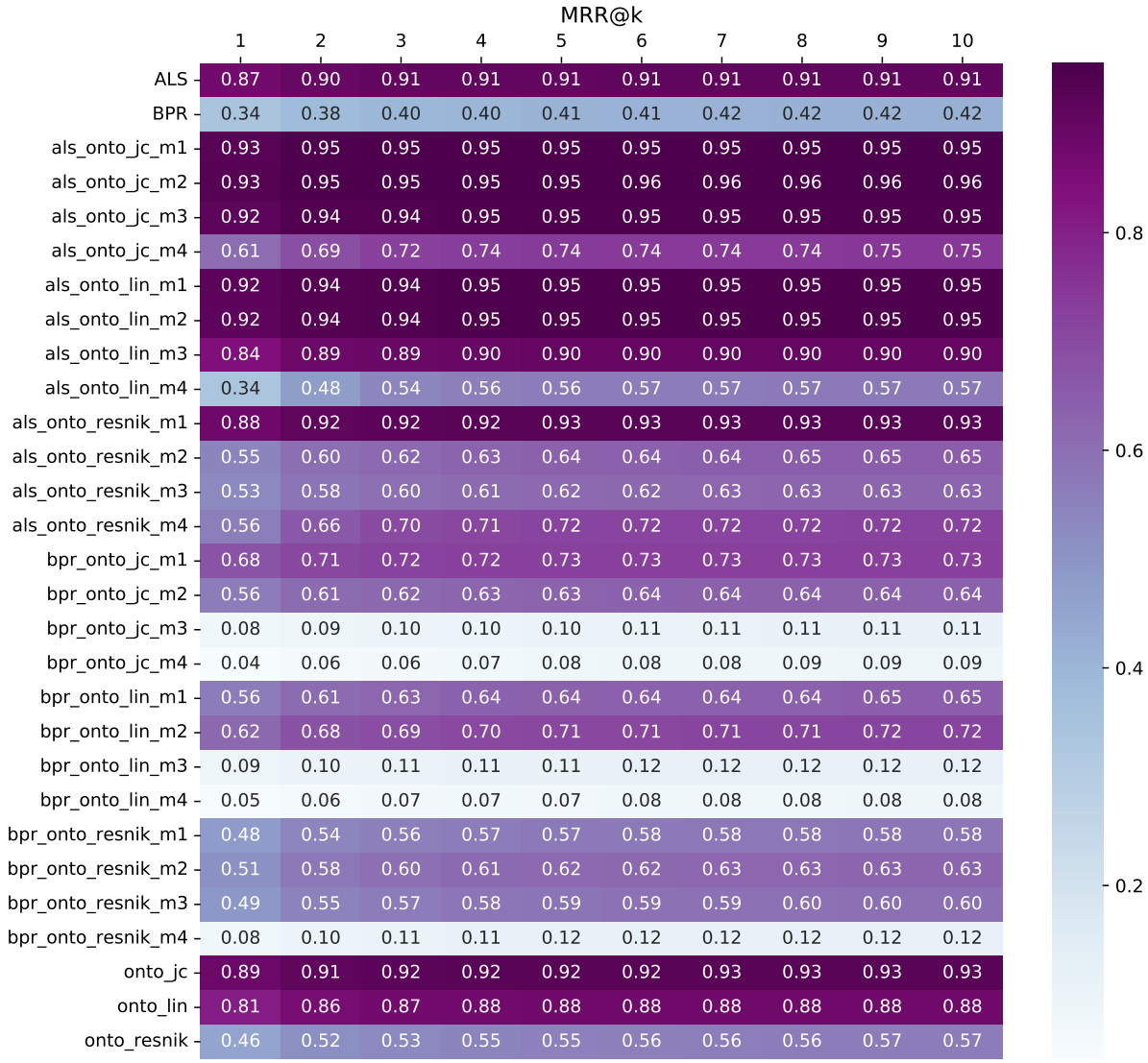


Figure A.35: A list of top@10 MRR results from Med4DB-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

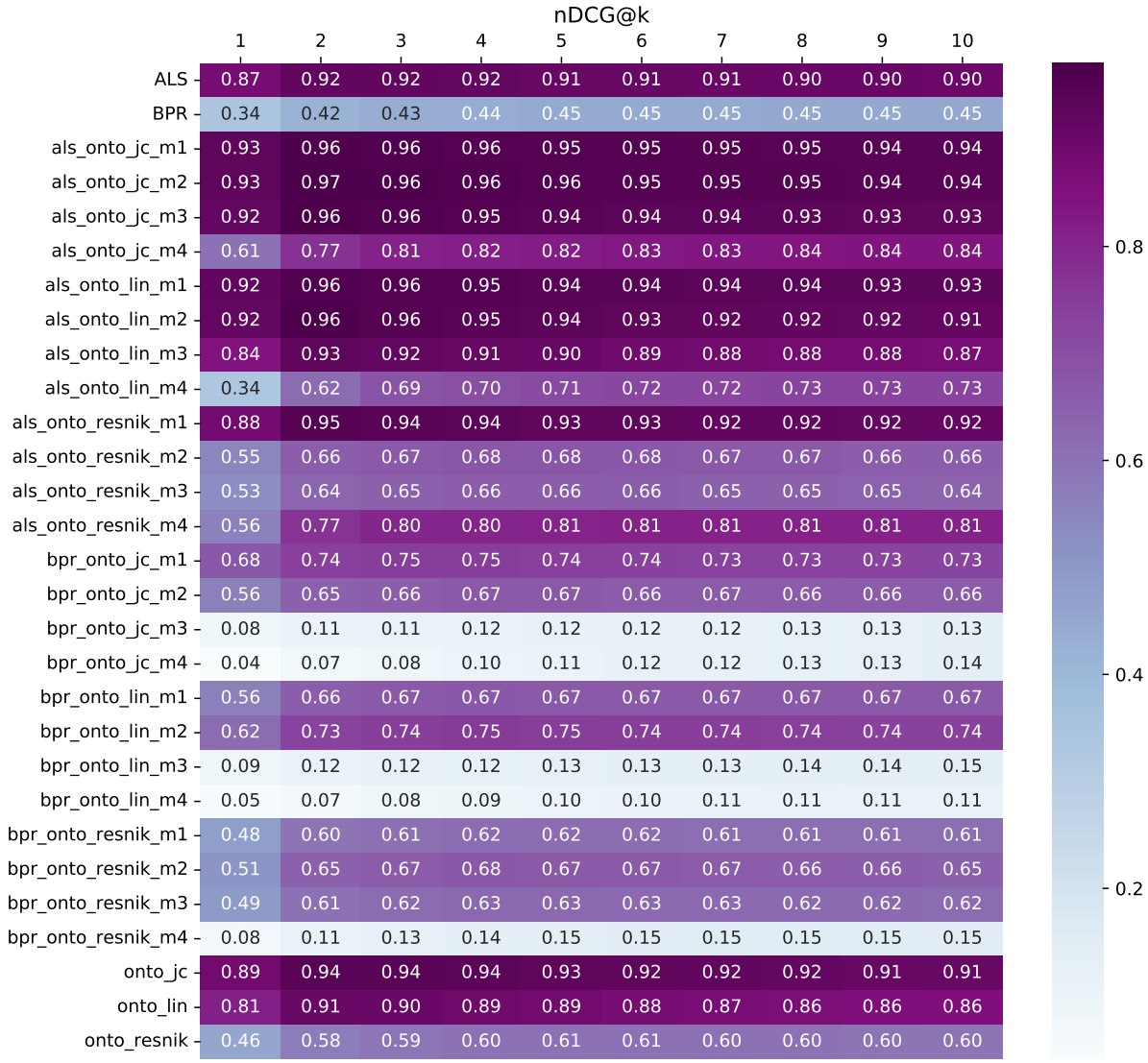


Figure A.36: A list of top@10 nDCG results from Med4DB-KG-Lin, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

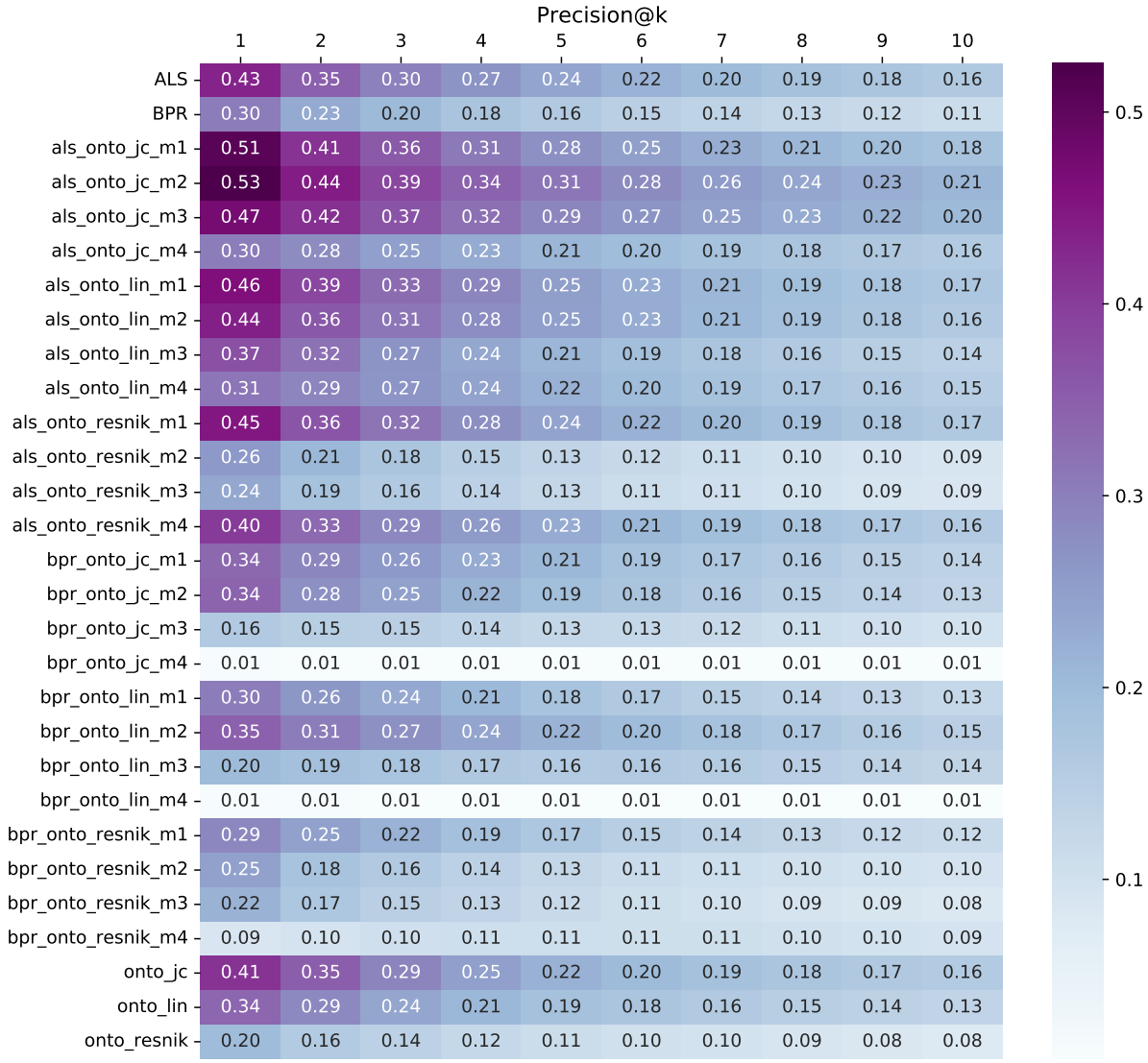


Figure A.37: A list of top@10 Precision results from Med4DB-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

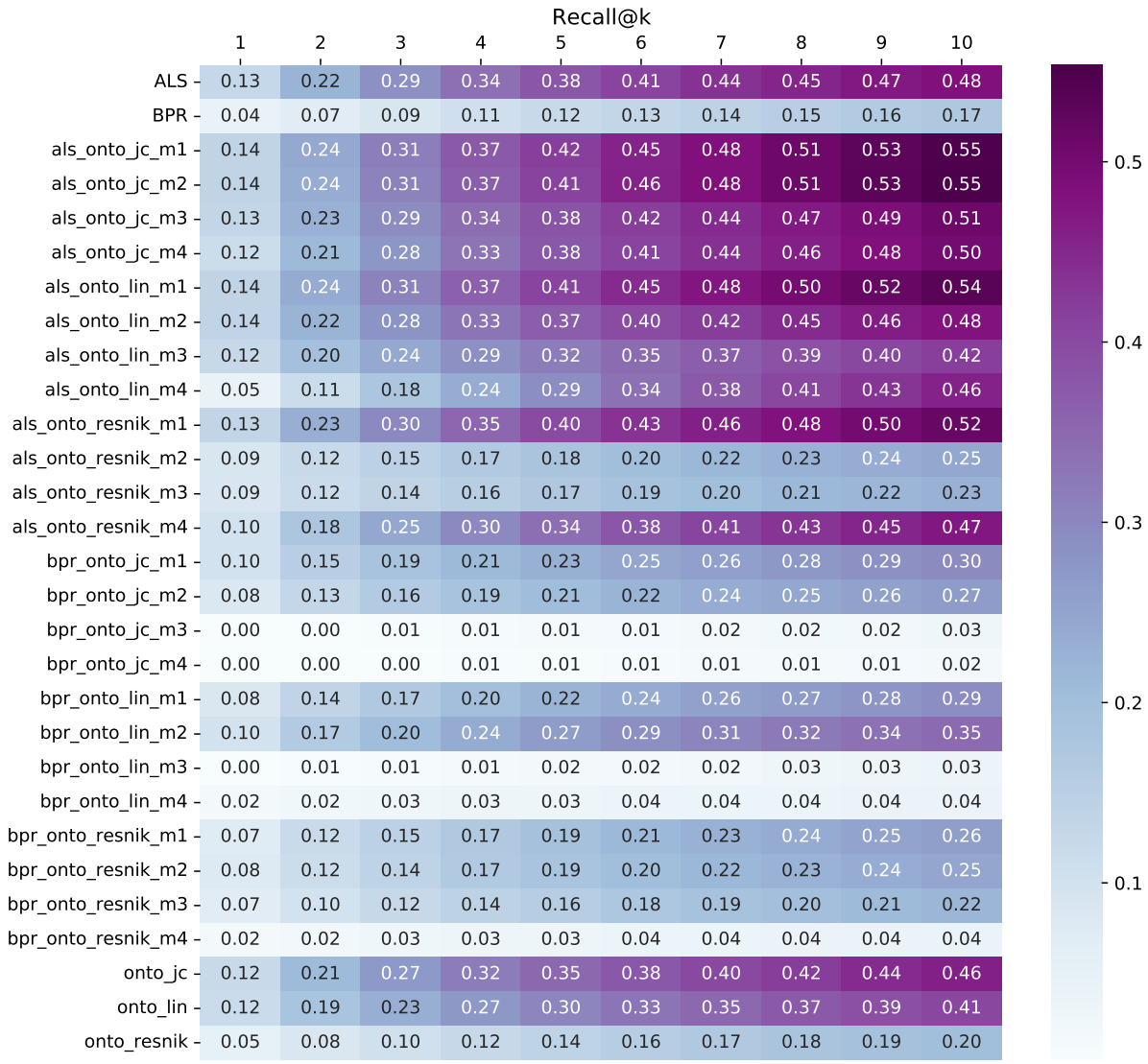


Figure A.38: A list of top@10 Recall results from Med4DB-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

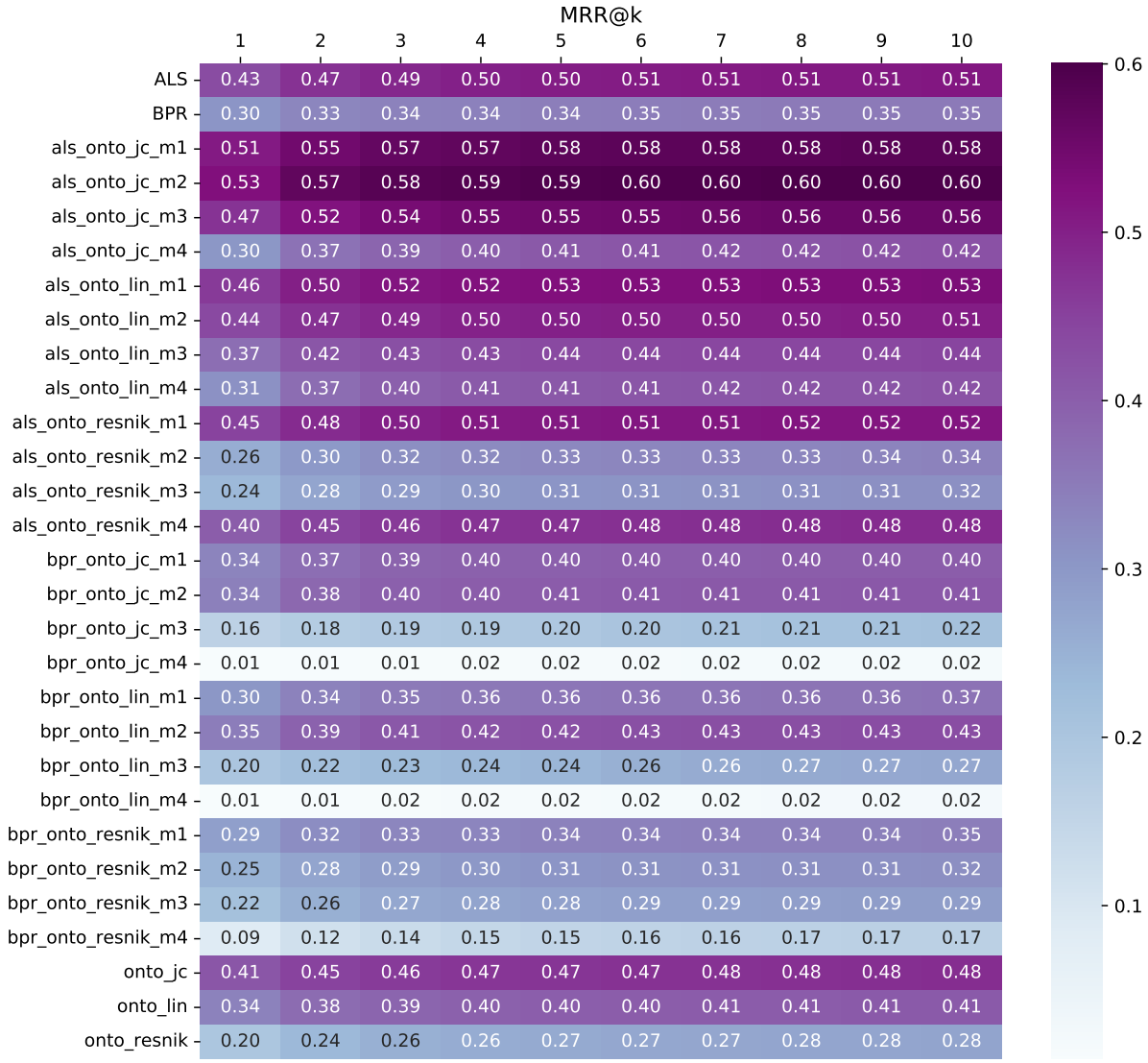


Figure A.39: A list of top@10 MRR results from Med4DB-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

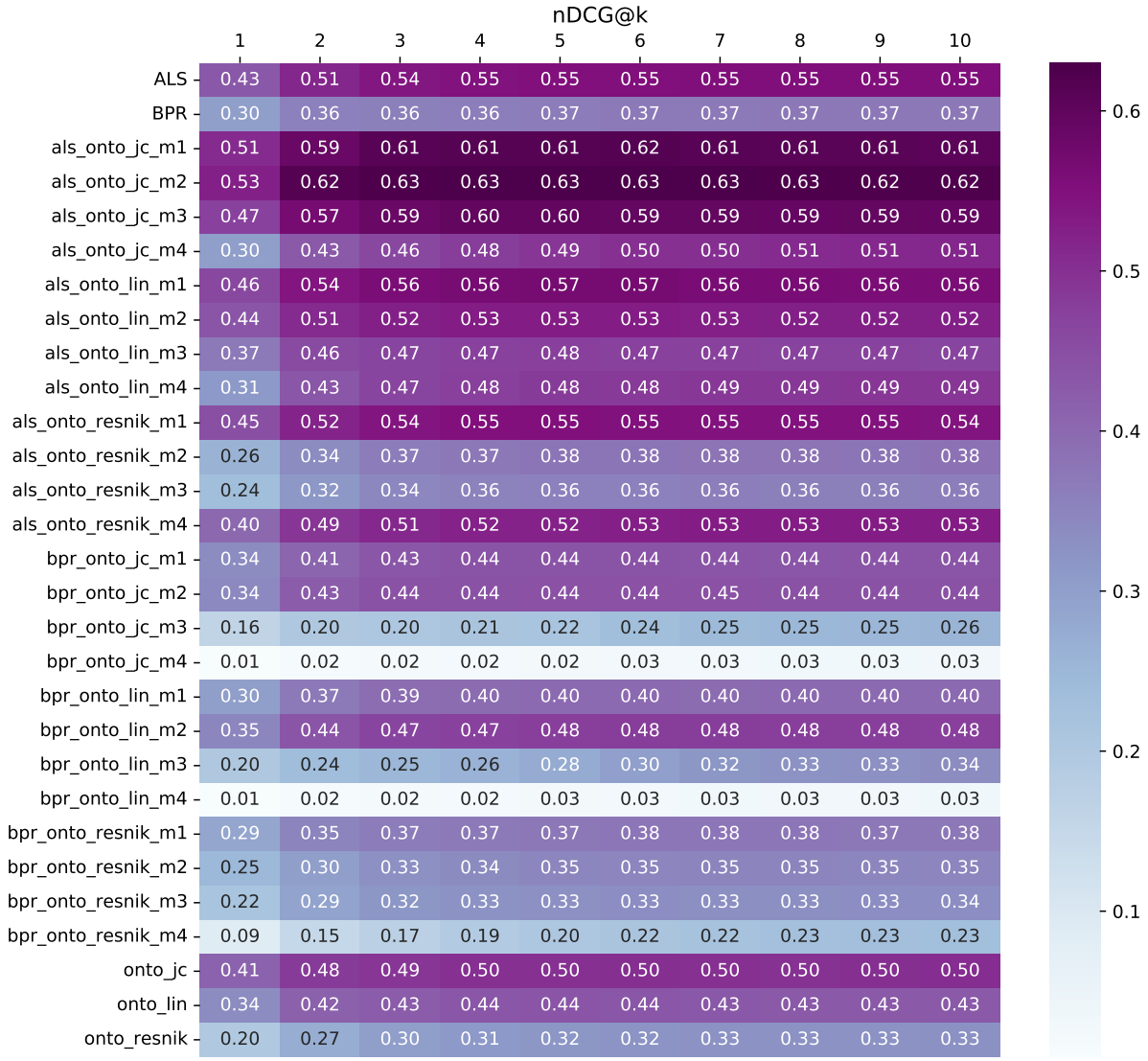


Figure A.40: A list of top@10 nDCG results from Med4DB-KG-Morgan, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

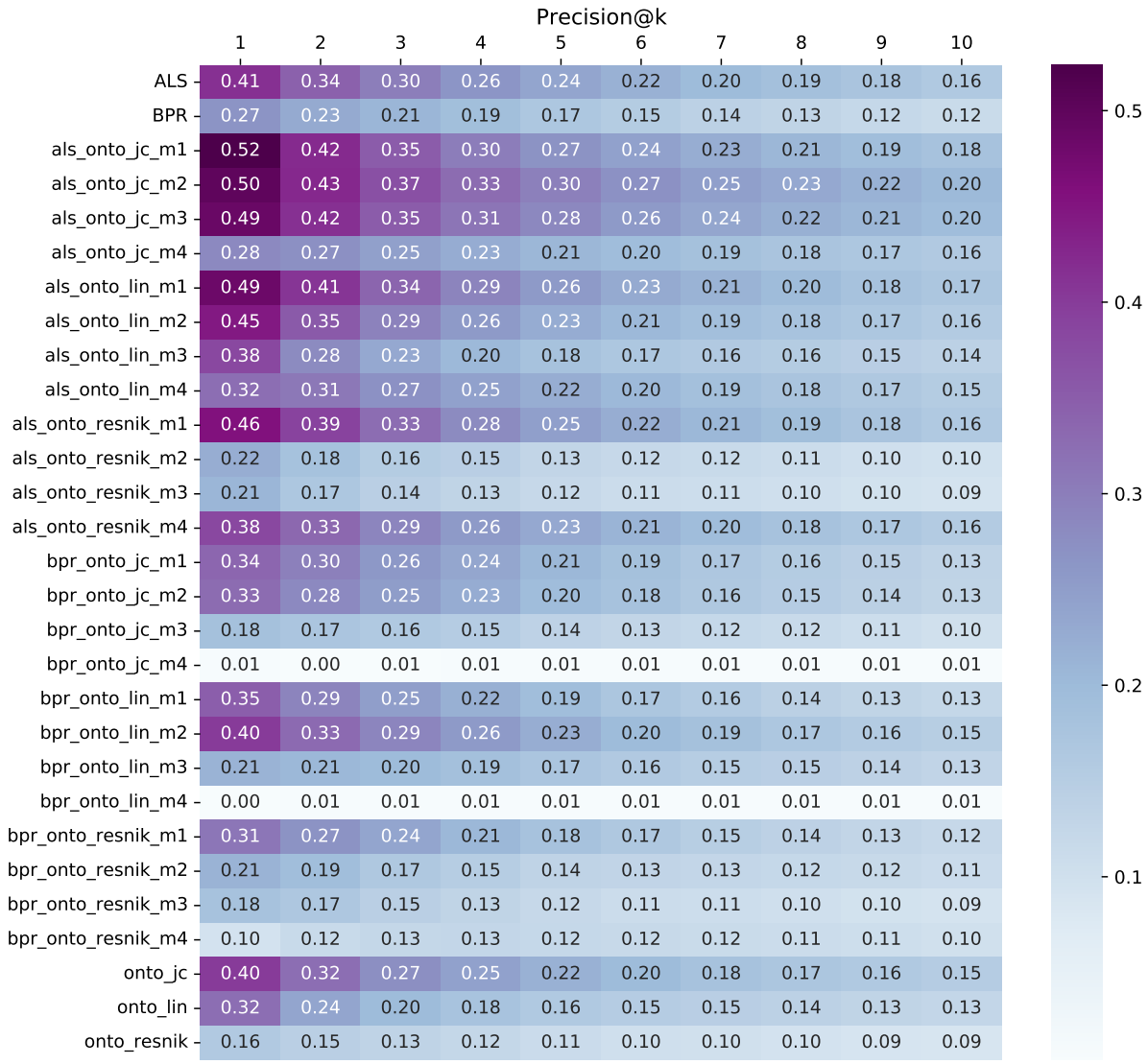


Figure A.41: A list of top@10 Precision results from Med4DB-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

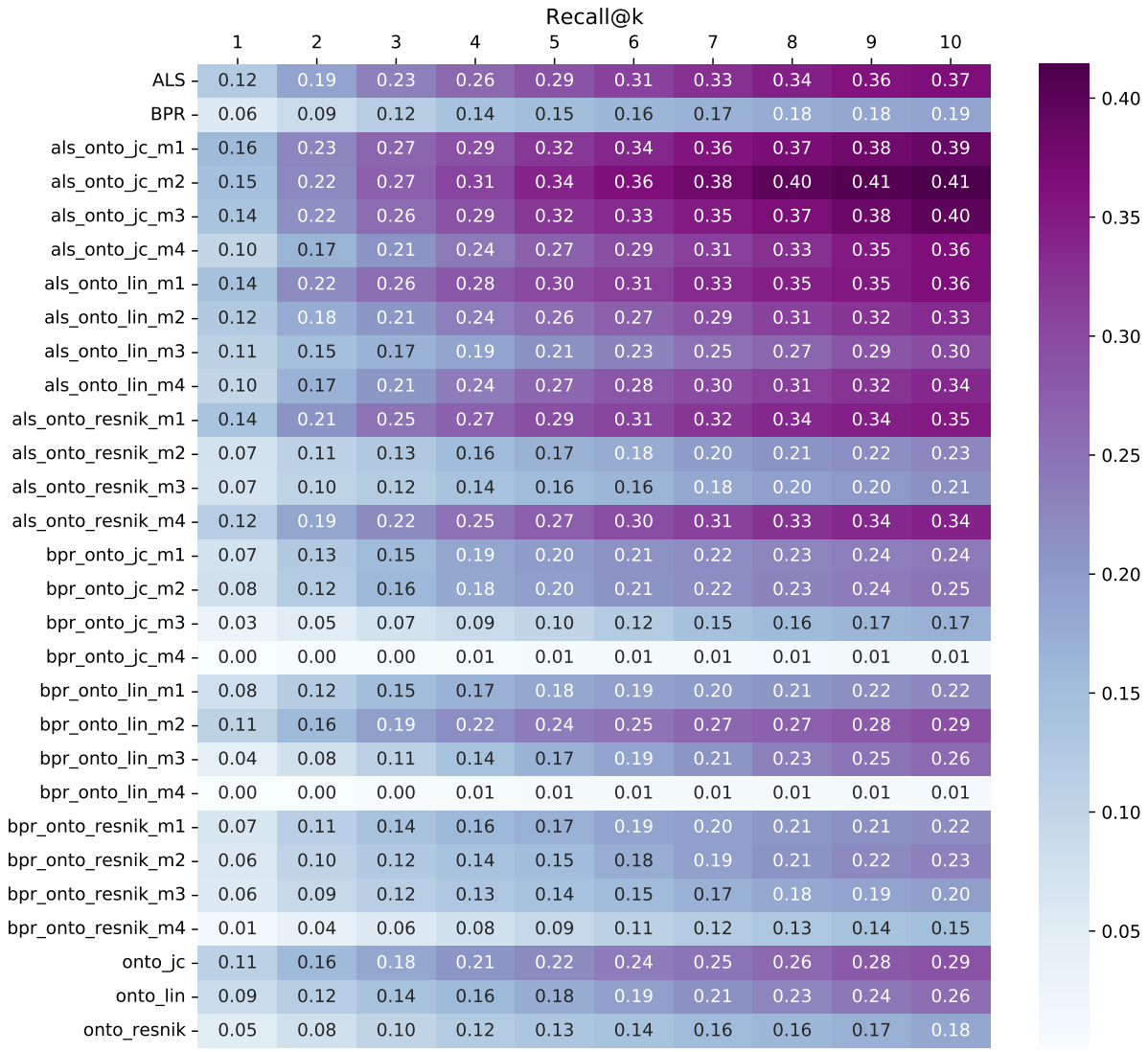


Figure A.42: A list of top@10 Recall results from Med4DB-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

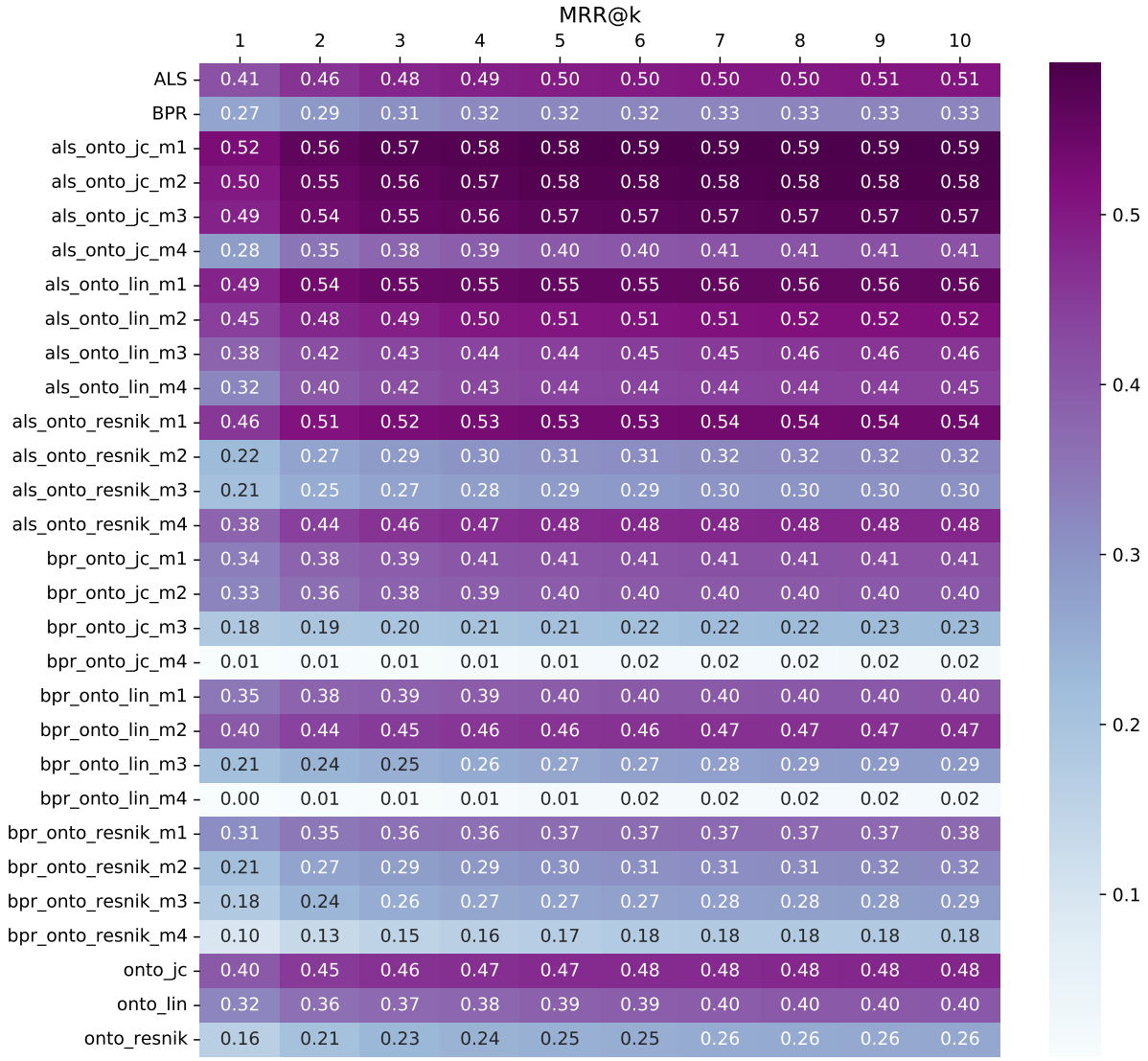


Figure A.43: A list of top@10 MRR results from Med4DB-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.

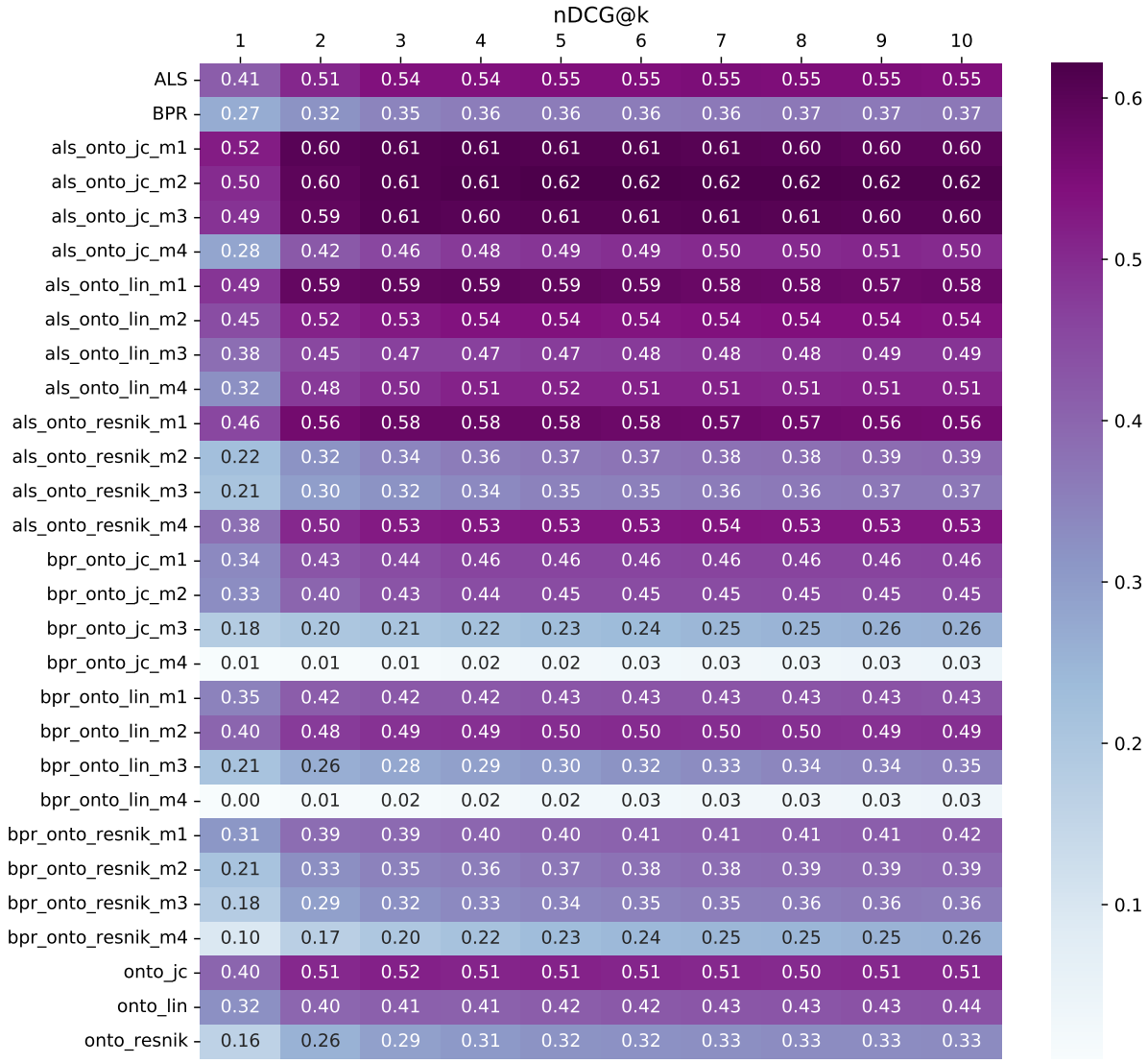


Figure A.44: A list of top@10 nDCG results from Med4DB-KG-Tanimoto, including ALS, BPR, ONTO, and hybrids, obtained using the 5 most similar items to calculate the ONTO scores.