Mathematical Information Retrieval based on Type Embeddings and Query Expansion

Yiannos A. Stathopoulos **Simone Teufel**
Computer Laboratory, University of Cambridge
15 JJ Thomson Avenue, Cambridge, UK
{yiannos.stathopoulos,simone.teufel}@cl.cam.ac.uk

Abstract

We present an approach to mathematical information retrieval (MIR) that exploits a special kind of technical terminology, referred to as a *mathematical type*. In this paper, we present and evaluate a type detection mechanism and show its positive effect on the retrieval of research-level mathematics. Our best model, which performs query expansion with a type-aware embedding space, strongly outperforms standard IR models with state-of-the-art query expansion (vector space-based and language modelling-based), on a relatively new corpus of research-level queries.

1 Introduction

Mathematical information retrieval (MIR) systems, such as MathWebSearch (Kohlhase and Prodescu, 2013; Hambasan et al., 2014), MIaS (Sojka and Líška, 2011) and Tangent (Pattaniyil and Zanibbi, 2014) have demonstrated that indexing and matching of formulae is beneficial to MIR. However, despite the sophistication of these methods, they leave elements of the natural language of mathematics largely unexploited. In contrast to general text, text in mathematics follows strong domain-specific conventions governing how content is presented (Ganesalingam, 2008). This is particularly so for research-level mathematics text (i.e., scientific articles), which is our main focus of interest. The conventionality of this text gives rise to many opportunities for the application of NLP to MIR. In this paper, we investigate the role of *mathematical types*, a special kind of technical terminology.

The term "type" refers to sequences of one or more words used to label mathematical objects (e.g., 'set", "smooth curve"), algebraic structures (e.g., "monoid", "group") and instantiable mathematical notions (e.g., "cardinality of a set"). Technical terms that are not used to refer to instances of these mathematical constructs are not types. Examples of non-types include references to the application of mathematical procedures (e.g., "proof by contradiction") and elements of the mathematical discourse (e.g., "theorem 4.1"). As a subclass of mathematical terminology, types are almost exclusively noun or prepositional phrases.

Types play a central role in communicating mathematical information by enabling mathematicians to name mathematical concepts, assign properties to objects and prove assertions about them. We consider types worthy of distinction from generic technical terms for two reasons: (a) types are used in the discourse to give sense to constituents of formulae and (b) types are used to consistently evoke mathematical concepts in textual argumentation and mathematical reasoning.

Our goal is to evaluate the usefulness of types as standalone lexical components in the retrieval of research-level mathematical information needs. Specifically, our hypothesis is twofold: (a) types are important discriminators in the mathematical discourse and (b) semantic relationships between types can be used to enrich queries and obtain significant improvements in retrieval efficiency.

A sub-task of our type-based approach is the construction of a type dictionary from a collection of documents. We address this in section 3 and describe a simple method for automatic identification of types. Furthermore, we evaluate our method using a gold standard set of type phrases which we have produced using judgements from 5 mathematicians.

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 2344–2355, Osaka, Japan, December 11-17 2016.

We focus on retrieving research-level mathematics because such material is rich in mathematical types. For instance, our system operates on queries such as the following (types are underlined):

> *Let P be a parabolic subgroup of $GL(n)$ with Levi decomposition $P = MN$, where N is the unipotent radical. Let p be an irreducible representation of $M(Z_p)$ inflated to $P(Z_p)$, how does $Ind_{P(Z_p)}^{GL_n(Z_p)}\pi$ decompose? It would be sufficient for me to know the result in the simplest case, where P is a Borel subgroup.*

Our experiments suggest that types are most effective when used to capture semantic relationships between mathematical concepts. Our top-performing type-based model, TypesExp, makes use of similarity in a type-aware word embedding space to identify semantically related types for query expansion. TypesExp outperforms state-of-the-art and traditional IR/query expansion models (described in section 4.3) demonstrating experimentally that types are valuable lexical components for IR in their own right (section 5).

2 Related Work

We use types to model mathematical concepts, but other constructs have been proposed in the literature for the same purpose. Grigore et al. (2009) take operators listed in OpenMath content dictionaries (CDs) to be mathematical concepts and use *term clusters* to model their semantics. A term cluster for a concept is composed of a label and a bag of nouns extracted from the operator description in the dictionary. This set is enriched manually using additional terms taken from online mathematical lexical resources. Grigore et al. assign the cluster that maximises the similarity (based on PMI and DICE) between the nouns in the local context of a target formula and those in the cluster, to represent the formula's semantics. Quoc et al. (2010) extract descriptions for formulae (phrases or sentences) from their surrounding context using a rule-based approach. Kristianto et al. (2012), on the other hand, used pattern matching on sentence parse trees and a "nearest noun" approach to extract descriptions, but these methods were later outperformed by SVMs (Kristianto et al., 2012).

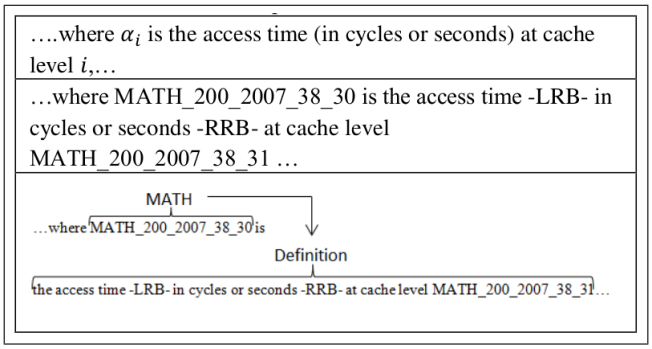

Figure 1: Example of extracting descriptions adapted from (Kristianto et al., 2012)

Our approach at modelling concepts with types incorporates the advantages of the described methods. Like term clusters, types label mathematical concepts and attach meaning in a distributional manner. However, rather than constructing type representations manually, we use types as concept labels and automatically compute distributional profiles for the concepts, based on word embeddings. Like formulae descriptions (Kristianto et al., 2012; Kristianto et al., 2014), our types are also extracted automatically from text. Although in some cases descriptions extracted by Kristianto et al. (2012, 2014) are noun phrases and resemble types, they often incorporate details that are particular to a specific context. For example, the extracted description for formula a_i presented in Figure 1 (adapted from (Kristianto et al., 2012)) contains information that is specific to the context (i.e., time at a particular cache level measured in cycles or seconds).

In our approach, we restrict types to relatively short technical terms which do not include context-specific information. This decision stems from our motivation to capture references to mathematical constructs in the form that they most consistently appear in scientific text.

3 Types for Math IR

Our definition of types is intended to model the perception of mathematical concepts shared between mathematicians and emerges from mathematical intuition. The notion of a type is intuitive to most mathematicians (as demonstrated in section 3.2). It is, however, hard to produce a concrete list of properties a technical term must adhere to in order to be considered a type.

Linguistically, types are a special kind of mathematical technical terminology. Technical terminology in general is well understood; for example, Justeson and Katz (1995) were among the first to define terminology as noun phrases with particular statistical properties. As lexical tokens, types are subject to a particularly high level of polysemy ("field" in Mathematics is a concept distinct to that in Physics) and synonymy (e.g., "karoubi envelope" is the same as "category of idempotent arrows"). Furthermore, many mathematical constructs are eponyms, i.e., named after their inventors, often additionally pre-modified by adjectives (e.g., "refined Noether normalization theorem", "abstract Hilbert space theorem"). New types can be formed through parameterisation (e.g.,"2-Group" and "G-Function" from "Group" and "Function" respectively) or by prepositional postmodification (e.g., "Ideal of a Ring", "Point on the Plane" and "Set of Matrices").

Conceptually, a type is any technical term that is (a) perceived by mathematicians to refer to mathematical objects, algebraic structures and mathematical notions and (b) can be instantiated in the discourse in the form of a variable. Here, we take mathematical objects to be anything that can be formally defined and manipulated in the discourse as part of formal deductive reasoning and/or proofs. This being said, it is important to highlight that some objects such as 'Number", "Matrix" and "Set" are considered basic and are never explicitly defined by mathematicians (Ganesalingam, 2008).

Like mathematical objects, algebraic structures also take part in mathematical manipulation but are defined as collections (or tuples) of other objects. Types can also refer to mathematical notions that can be instantiated as variables or can take the form of other objects. For example, an "envelope of elliptic trajectories" can be an "ellipse". Named axioms, theorems and conjectures are also considered to be types since they refer to universally accepted, formally defined constructs for the purpose of argumentation (e.g., to complete a proof).

Any technical term that does not fit the above description is not a type. Examples of non-types include properties of operators (e.g., "Associativity"), mathematical procedures (e.g., "Proof by contradiction"), processes (e.g., "Differentiation") and theories (such as "Chaos"). Note that mentions of mathematical theories are ambiguous: it is often unclear whether they refer to a branch of mathematics or to a formally defined construct. These examples are not types because either (a) they are not explicitly referring to mathematical constructions (i.e., they cannot assign meaning to variables), (b) their role in the discourse is indirect (e.g., properties capture relationships between concepts) and (c) they are used as discourse labels for anaphoric purposes.

Types can be organised hierarchically, with some types being specialised instances of other, more abstract constructions. This relationship is often mirrored linguistically: a type expressed by a longer string is often a subtype of the type corresponding to a sub-sequence string. For example, a "smooth curve" is a subtype of "curve". We consider sub-types to be distinct atomic units with discrete meaning despite the fact that they are constructed linguistically in a compositional manner with regards to their super-type. However, not all type/sub-type relationships are expressed on the surface. For example, it is not obvious that "Klein Bottle" is a sub-type of "Surface".

Types are relevant to both textual and mathematical contexts of queries and documents. As a result, types have potentially more impact than generic technical terms. Given that types communicate ideas shared between mathematicians, we anticipate that modelling the distributional profile of types (e.g., using an embedding space) will be beneficial to MIR.

3.1 Automatic Type Detection and Extraction

We address the problem of automatic detection and extraction of types so that we can perform large-scale experimentation. Our method proceeds as follows. First, we use the C-Value algorithm (Frantzi et al., 1998) on our corpus to extract technical terms (candidate types).

Given a collection of documents as input, the C-Value method identifies multi-word technical terms using both a linguistic and a statistical component. The linguistic component is employed primarily for eliminating multi-word strings that are unlikely to be technical terms. This is done by enforcing a stop-word list (high-frequency corpus terms) and through the application of linguistic filters (regular expressions) on sequences of part-of-speech tags. The statistical component assigns a "termhood" score to a candidate sequence based on its corpus-wide statistical characteristics and those of the sequences that contain it.

Each entry in the output of the C-Value algorithm corresponds to one technical term – an equivalence class of all variations of the term in the corpus. In the next step, our process selects technical terms that are likely to be types. We assume that technical terms that are types have an entry in the Encyclopedia of Mathematics[1] (8730 articles in total at the time of download) and/or an entry in Wikipedia (we used a 2014 Wikipedia dump) under the key categories "mathematical objects", "mathematical concepts" and "mathematical structures" and their sub-categories.

A dictionary of types is constructed by including technical terms that entirely match the title of one or more of these encyclopedia articles. We have opted to use this intersection of technical terms and article titles, as opposed to the titles alone, because not all titles are useful. For example, although the Wikipedia article[2] "The geometry and topology of three-manifolds" is filed under the categories of interest "Hyperbolic geometry", "3-manifolds" and "Kleinian groups", the title as a whole does not represent a single concept. In contrast, the technical term "Riemannian manifold" would completely match the title of the Wikipedia[3] (or Encyclopedia of Math) article for the concept and would thus be identified as a type by our method. The application of our method to the Mathematical REtrieval Corpus (MREC) (Líška et al., 2011)[4] has produced a dictionary of 10601 types.

3.2 Gold Standard Evaluation

Our definition of types is intricate and requires mathematical expertise. We also expect it to be subjective. We therefore evaluated the quality of accumulated types using 5 judges (third-year undergraduate and graduate mathematicians). Participants were shown a mixed list of types (as determined by our system) and non-types (technical terms that had been filtered out by our method as non-types). Without knowing what the source of each type was, they were asked to identify types using a 2-page definition of types (16 rules). The technical term list they were asked to judge consisted of 200 phrases and was constructed as follows: (1) two-thirds of the sample are sourced from the type list (10601 phrases). This set of 134 phrases is produced by sampling by observed distribution over phrase length. (2) The remainder of the sample (66 phrases) is sourced from the original list of C-Value technical terms not identified as types by our method. The termhood scores of these technical terms can range from very high to very low. As a result, we split the original technical term list (2.8 million phrases) into three equally-sized segments based on their C-Value score: (a) high score, (b) medium and (c) low score. From each segment, we removed any term already identified as a type and drew 22 phrases from the remainder. As before, segment samples are sampled by observed distribution based on phrase length. Sampling negatives from the three segments, as described above, enables us to compare the spread of positives and negatives across C-Value scores. (3) The two parts of the sample are concatenated and shuffled randomly. An HTML questionnaire is automatically produced and presented to annotators.

Precision and recall of our type identification method was $P = 73.9\%$ and $R = 81.8\%$ respectively, resulting in an F-score of 77.7%, with respect to the majority opinion. As expected, this is a subjective task, and without any specific training, annotator agreement, measured using Fleiss's Kappa (Fleiss, 1971), is in an intermediate range ($K = 0.65; N = 200, k = 2, n = 5$). We observed that judges often judged the following technical terms as types, although this contradicts our definition: author names, mathematical properties and non-sensical phrases.

[1] https://www.encyclopediaofmath.org

[2] https://en.wikipedia.org/wiki/The_geometry_and_topology_of_three-manifolds

[3] https://en.wikipedia.org/wiki/Riemannian_manifold

[4] The MREC is a subset of ArXiv and is composed of over 439,000 scientific papers which have had all LaTeX formulae converted into MathML (Líška et al., 2011). It is the document collection underlying the test collection we use (section 4).

4 IR Experiments

Traditional retrieval models operate under the assumption that each constituent term in a multi-word type is an independent source of information. Our intuition is that the words in multi-word types carry more information as a group and should therefore be treated as atomic lexical units by these models. We propose two type-aware models, based on the traditional Vector-space model (VSM). In our evaluation we compare our type-aware models to established traditional, term-based[5] retrieval models. The motivation behind this approach is to clearly identify the effects of type information. We keep the comparison at the lexical level so that the effects of types to retrieval performance can be isolated – models employing formulae indexing and matching are not considered.

4.1 Computing a Type Embedding Space

We use our dictionary of types (section 3.1) to construct a type embedding space – a word embedding space that includes embeddings for types as atomic lexical units. The type embedding space is used to assign meaning (or denotation) to types in the form of vector embeddings and to expand queries with new types (section 4.3.1).

The type embedding space is constructed as follows. First, we apply sentence tokenisation over the MREC using the Stanford CoreNLP toolkit (Manning et al., 2014). Subsequently, we apply longest sequence matching on the words of each sentence and identify all instances of types in the the corpus. Once detected, word sequences belonging to types are replaced by a single token (a concatenation of the constituent words of the type). As we are interested in modelling the relatedness of meaningful linguistic tokens, rather than mathematical artefacts, we replace MathML blocks representing formulae in the sentence by a single token ("@@@"). Finally, the re-written sentences are passed on to `word2vec` in skipgram mode (Mikolov et al., 2013a; Mikolov et al., 2013b) with negative sampling and window size=10 to produce the type embedding space.

4.2 Test Collection

Evaluation is carried out using the Cambridge University MathIR Test Collection (CUMTC) (Stathopoulos and Teufel, 2015), which is composed of 120 real-life MIR topics procured from the MathOverflow (MO) on-line community. As illustrated in Table 1, each MO thread in the CUMTC is sentence-tokenized, with sentences either being part of the "prelude" (introduction to the mathematical subject of interest) or part of a concrete sub-question. We produced 160 queries from the CUMTC by emitting one query per sub-question in the collection. The body of each query is obtained by concatenating the text of the sub-question to the text of the associated prelude.

Prelude	Let P be a parabolic subgroup of $GL(n)$ with Levi decomposition $P = MN$, where N is the unipotent radical. It would be sufficient for me to know the result in the simplest case, where P is a Borel subgroup.
SQ-1	Let p be an irreducible representation of $M(Z_p)$ inflated to $P(Z_p)$, how does $Ind_{P(Z_p)}^{GL_n(Z_p)}\pi$ decompose? (at least until $g = 3$).

Table 1: Topic 175 (MO post 90038), prelude and sub-question

We chose this test collection because its topics represent real-life, research-level information needs, expressed in the natural language of mathematics, and are rich in mathematical types. This is in contrast to the NTCIR (Aizawa et al., 2013; Aizawa et al., 2014) test collections which emphasise formulae search (Guidi and Coen, 2015) with queries primarily taking the form of bags of keywords and formulae. Although the NTCIR Open Information Retrieval (OIR) is composed of free-text queries like those in the CUMTC, these are not accompanied by pre-determined relevance judgements[6]. Furthermore, textual descriptions in the OIR evaluation set are not as linguistically rich as the text in mathematics papers. Documents and queries are processed using a single pipeline that performs case normalisation and employs the Tika framework to flatten MathML consistently.

[5] A bigram model has also been considered but excluded from the comparison due to extremely poor performance.

[6] Relevance is judged using interactive sessions with humans.

4.3 Experimental Design

Performance is measured using mean average precision (MAP). Queries derived from the CUMTC are fairly long (averaging 88 words) and describe highly specialised information needs. As a result, they have relatively few relevant documents (only 19.17% have more than 1 relevant documents). Thus, our experimental setup is closer to that of the TREC HARD and TREC Robust tasks (Voorhees and Harman, 2005), than to more general-purpose retrieval (such as NTCIR and TREC ad-hoc) and low MAP values are to be expected. In the case of the CUMTC, the sparsity of relevant documents is attributable to the difficulty of the queries, rather than to the inherent uniqueness of the answer (as is the case in homepage search). One of the effects of the nature of the CUMTC is that the MAP scores of IR models will be numerically low. However, note that the small number of relevant documents per query does not make the evaluation per se unstable: we use a large number of queries and adopt the paired permutation test for significance testing. The permutation test is a non-parametric test for mean difference and is known to be reliable with MAP and its derivatives (Smucker et al., 2007).

In order to investigate the usefulness of mathematical types for retrieving research-level mathematics we adopt a two-level comparison of retrieval models. On one level, we compare traditional, term-based retrieval models to type-aware derivatives (see Figure 2 for the derivational relationship between models). On the second level, we compare the effectiveness of term-based query expansion to that based on types.

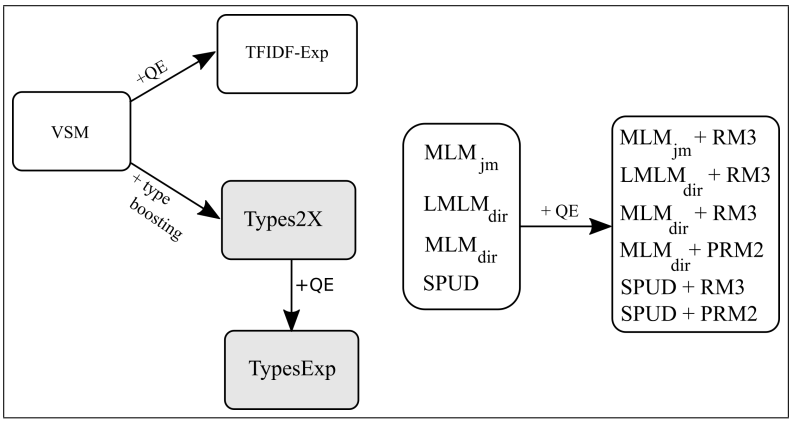

Figure 2: Relationships between basic models and their more sophisticated derivatives (grey boxes represent type-based models, white boxes represent term-based models).

The first level of comparison is performed across IR paradigms and includes term-based models that employ heuristic methods (e.g., VSM and BM25) as well as language modelling (e.g., classical multinomial language model (Zhai and Lafferty, 2001)). At this level, we wish to (a) determine the performance of traditional IR models on the CUMTC and (b) investigate if types are more useful than simple terms when retrieving research-level mathematics. A break-down of considered models based on this two-level comparison is presented in Table 2.

	No Expansion	**Query Expansion**
Terms	VSM	TFIDF-Exp
	BM25	-
	MLM_{jm}	MLM_{jm}+RM3
	MLM_{dir}	MLM_{dir}+RM3, MLM_{dir}+PRM2
	$LMLM_{dir}$	$LMLM_{dir}$+RM3
	SPUD	SPUD+RM3, SPUD+PRM2
Types	Types2X	TypesExp

Table 2: Overview of models.

4.3.1 Retrieval Models

We propose and evaluate two type-based heuristic models, based on VSM, that assign elevated significance to types through 2X boosting. One of our models makes use of inter-type similarity, encoded in a type embedding space, to expand queries.

Lucene VSM with 2X type boosting (Types2X). We apply the boosting pipeline described by Stathopoulos and Teufel (2015) to our type dictionary (section 3.1); i.e, we apply longest matching to a Lucene positional index and emit a type-aware "delta index". This type discovery and normalisation pre-processing step is also applied to queries. Our assumption is that types are a valuable source of information for an MIR system. Types2X assumes the role of a type-aware model that performs the simplest manipulation of those types that are physically present in the query (simple 2X boosting). Therefore, in our comparison, Types2X is used to measure the simplest possible way of incorporating types during retrieval (as opposed to just using terms). We expect Types2X to perform better than Lucene VSM (the term-based model it is based on) but no better than models incorporating type information in more sophisticated ways (such as TypesExp).

Lucene VSM with 2X type boosting and type-based query expansion (TypesExp). Unlike mathematical papers, queries are not always rich in types: on average each query contains around 13 type instances while documents in the MREC contain on average close to 548 type instances. TypesExp overcomes this problem by enriching queries with types. Queries are expanded using the types (as opposed to the terms) they contain. For each type in a query, the type-embedding space (using `word2vec` similarity as discussed in section 4.1) is used to discover n fresh related types. Semantic relatedness between types is modelled by the cosine similarity of their vector representations. The set of fresh types is appended to the original query and the new query is executed on a 2X type up-weighted VSM. The value for n is the only parameter of the model, which has been experimentally set to $n = 5$.

4.3.2 Baseline Models

Traditional, term-based retrieval and query expansion models [7] are used as baselines so that the effects of a-priori knowledge of types to retrieval performance can be isolated and quantified against uninformed approaches. Two query expansion (QE) methods based on pseudo-relevance feedback (PRF) are considered. The first method, known as RM3 (Abdul-jaleel et al., 2004; Lv and Zhai, 2009), assumes that documents and queries are generated by the same relevance model. The second, referred to as PRM2 (Lv and Zhai, 2011), makes use of proximity information in the feedback documents to expand queries.

Simple Baselines We use Lucene's default VSM (based on cosine similarity) and BM25 (Harter, 1975; Robertson et al., 1981; Robertson and Walker, 1994; Robertson et al., 1994) implementations. These models are considered not because of their strength per se, but because of their usefulness in identifying the effects of types in the performance of heuristic models: they are lingustically uninformed models. Furthermore, they are useful in quantifying the effects of boosting query types alone (Types2X).

Language Models (LM) Smoothed instances of the classical multinomial language model (Zhai and Lafferty, 2001) are also considered. The MLM_{jm} multinomial model employs Jelinek-Mercer (JM) smoothing. We rely on Lucene's implementation of the model and, in the absence of training data for the parameter λ, we use $\lambda = 0.7$ since there is strong evidence that this value is optimal for long queries (Zhai and Lafferty, 2001; Zhai and Lafferty, 2004). Two implementations of the multinomial model with Dirichlet smoothing are used in our comparison: the Lucene implementation ($LMLM_{dir}$) and the implementation by Cummins et al. (2015) (MLM_{dir})[8]. The smoothing parameter, μ, is set to $\mu = 2000$, which Zhai and Lafferty (2001, 2004) found to be near-optimal for large queries, such as those in our setup. These language models are included as more sophisticated, type-agnostic alternatives to the basic heuristic models.

[7] Technical issues have prevented us from incorporating the formula-aware Tangent and MiAS models in our evaluation.

[8] In correspondence with one of the authors of Cummins et al. (2015), it has come to our attention that there is suspicion in the community that the Lucene implementation of the classical multinomial language model may be incorrect for large queries, so we report two implementations for safety.

SPUD SPUD is a state-of-the-art unigram LM that models documents as draws from a multivariate Polya distribution (Cummins et al., 2015). Smoothing of the document model is performed using a linear combination of the unsmoothed document and background models and is controlled through a smoothing parameter ω. The SPUD ranking method estimates the probability that a query is generated from the expected multinomial drawn from each document model. We use the SPUD implementation by Cummins et al. (2015) with default parameters ($\omega = 0.8$), which have been shown to produce good results with long queries (Cummins et al., 2015). SPUD is included in our comparison as a state-of-the-art type-agnostic LM baseline against which we can benchmark the performance of our type-based models.

We also consider versions of the stated language models augmented with query expansion methods (RM3 and PRM2) implemented as part of the code accompanying (Cummins et al., 2015)[9]. Parameter values for these query expansion models are also taken from Cummins et al. (2015). LMs with QE are used to determine how type-based QE performs in comparison to state-of-the-art term-based QE.

Automatic QE using top TF-IDF terms (TFIDF-Exp). In this model, the query is expanded using the top s terms in the query as determined by document collection-wide TF-IDF scores. Stopwords and words with term frequency lower than 50 are excluded. Like before, each selected term is expanded using its n-nearest neighbours in a word embedding space (skip-gram, window size =10). The model has two parameters: *pool size* (n) and *seed size* (s). We found experimentally that the model performs best for $n = 1$ and $s = 1$. This baseline is used to determine whether any performance improvements obtained by type-based QE using an embedding space are due to types, rather than the use of embedding spaces in general.

5 Results and Discussion

Table 3 shows the results of all retrieval models considered[10]. The last two columns indicate the significance in MAP difference between each model and our proposed type-based models. As expected, absolute MAPs are low across the board, a phenomenon that can be attributed to the complexity of the underlying information needs and the resulting small number of relevant documents per query. But as long as the evaluation is stable, it is only the comparative performance of each model that we should be interested in.

	VSM	BM25	MLM_{jm}	$LMLM_{dir}$	MLM_{dir}	SPUD	TF-IDFExp	MLM_{jm} +RM3
MAP	.076	.079	.084	.072	.066	.090	.060	.063
TypesExp	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$
Types2X	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$
	$LMLM_{dir}$ +RM3	MLM_{dir} +RM3	MLM_{dir} +PRM2	SPUD +RM3	SPUD +PRM2	Types2X	TypesExp	
MAP	.051	.082	.061	.050	.072	.094	.150	
TypesExp	$\gg$	$\gg$	$\gg$	$\gg$	$\gg$	$>$	-	
Types2X	$>$	$\approx$	$\approx$	$>$	$\approx$	-	$<$	

Table 3: Model MAP performance and comparison to TypesExp and Types2X models.

The model utilising type-based expansion (TypesExp) is our best model; it outperforms every other model. In some cases the differences are dramatic; TypesExp's MAP score is twice that of the Lucene VSM and, to the best of our knowledge, 0.15 MAP represents the state-of-the-art on the CUMTC. We now discuss which of TypesExp's components contributed most to this improvement over existing IR models. Although the up-weighting of types on its own (Types2X) improves MAP over the VSM, the difference in performance is not statistically significant. This is also the case when comparing type up-weighting to the majority of alternative models. The best performing model not employing query expansion is SPUD, followed closely by MLM with JM smoothing. The traditional BM25, VSM and

[9] https://github.com/ronancummins/spud

[10] In Tables 3, 4 and 5, $\ll$ indicates that "column" is significantly better than "row" $\alpha = 0.01$; $<$ at $\alpha = 0.05$. $\approx$ indicates that difference is not significant.

Lucene MLM_{dir} models performed comparably, with no significant differences in MAP between them (as observed in Table 4).

VSM	-						
BM25	$\approx$	-					
MLM_{jm}	$\approx$	$\approx$	-				
$LMLM_{dir}$	$\approx$	$\approx$	$\approx$	-			
MLM_{dir}	$\approx$	$\approx$	$\approx$	$\approx$	-		
TFIDF-Exp	$\approx$	$\approx$	$\approx$	$\approx$	$\approx$	-	
SPUD	$\approx$	$\approx$	$\approx$	$\approx$	$>$	$\approx$	-
	VSM	BM25	MLM_{jm}	$LMLM_{dir}$	MLM_{dir}	TFIDF-Exp	SPUD

Table 4: Comparison of models not employing query expansion.

	MLM_{jm} +RM3	$LMLM_{dir}$ +RM3	MLM_{dir} +RM3	MLM_{dir} +PRM2	SPUD +RM3	SPUD +PRM2
MLM_{jm}	$>$ **.021**	$>$ **.033**	.002	.023	.034	.013
MLM_{dir}	.003	.015	.016	.005	.016	.006
$LMLM_{dir}$	.009	$\gg$ **.021**	.01	.011	.022	.001
SPUD	$>$ **.028**	$>$ **.04**	.009	$>$ **.029**	$\gg$ **.04**	.019

Table 5: Absolute difference in MAP of unexpanded and query-expanded models.

General-purpose query expansion methods appear to be ineffective in retrieving research-level mathematics. From the results in Table 5 we observe that versions of the models augmented with state-of-the-art query expansion are either significantly outperformed or not significantly better than their corresponding basic versions. In other cases, the vanilla models significantly outperform their query expanding counterparts (e.g.,MLM_{dir} and $MLM_{dir} + RM3$, $SPUD$ and $SPUD + RM3$). This is in contrast to type-based query expansion, where the TypesExp model (2X type up-weighting+expansion using a type-aware embedding space) significantly outperforms both the VSM and Types2X models it is based on. The observations described above seem to point to the fact that, in the context of MIR, mathematical types encode more information than the sum of their individual, constituent terms.

On one hand, the performance of Types2X suggests that information coming from the types occurring in the queries alone may not be enough to produce significant improvements in retrieval efficiency. On the other hand, our experiments have shown that it is only the combination of query expansion with type information (rather than with simple terms) that yields significant performance gains on these difficult queries. Our intuition is that type-based expansion introduces semantically related concepts that elevate the score of topically relevant documents. Insight into why this method performs well can be obtained by looking into how types are expanded. The query in Table 6 is topic 175 (MO post 90038[11]). From an initial set of 6 types in the dictionary, our method expanded the query with 14 more types.

Query	Let P be a *parabolic subgroup* of $GL(n)$ with *Levi decomposition* $P = MN$, where N is the *unipotent radical*. Let p be an *irreducible representation* of $M(Z_p)$ inflated to $P(Z_p)$, how does $Ind_{P(Z_p)}^{GL_n(Z_p)}\pi$ decompose? It would be sufficient for me to know the result in the simplest case, where P is a *Borel subgroup*.
Query types	'levi decomposition', 'parabolic subgroup', 'unipotent', 'irreducible representation','borel subgroup'
Added Types	'reducible representation', 'regular element', 'conjugacy class', 'iwasawa decomposition', 'unipotent element', 'cartan decomposition', 'finite-dimensional representation', 'unitary representation','cartan subgroup', 'centralizer', 'subgroup', 'triangular decomposition', 'jordan decomposition','parabolic subalgebra'
TFIDF-Exp terms	'nilpotent', 'shredded', 'semi-simple', 'inflates', 'centralizer', 'pro-', 'puffed','engulfed', 'inflate', 'semisimple'

Table 6: Topic 175 (MO post 90038): Text and types in query, types in query and dictionary, expansion types and sample TFIDF-Exp expansion terms (n=5, $s = 2$).

Broadly speaking, all of the types expanded by our system are topically related to the types in the query. The types "iwasawa decomposition" and "cartan decomposition" in the expanded set strongly

[11] http://mathoverflow.net/questions/90038

relate to the types in the query (both are related to Levi decomposition of Lie algebras). However, there are also some instances of weak association between query and expanded types. For example, the type "subgroup" is almost certainly too general to do any good. On the other hand, the TFIDF-Exp expansion set for this query (bottom row of table 6) appears to be less likely to contain terms related to the topic subject. The "bag-of-types" model (Types2X) is inhibited by the fact that queries have a much smaller vocabulary of types compared to mathematical documents. Furthermore, we hypothesise that the model's limited performance and the effectiveness of topical relationships discovered through types and type embeddings are attributable to three characteristics of the mathematical discourse.

First, types can share algebraic structure, which permits mathematicians to perform mathematical reasoning by manipulating the shared components and properties. For example, both "monoid" and "semi group" have an underlying "set" and an associative binary operator. Thus, mathematics text may coerce either structure to its carrier set in formal argumentation and make statements that are true for both structures (Ganesalingam, 2008). Second, phenomena like polysemy and synonymy are frequent in mathematics. Lexically distinct terms can end up referring to the same concept: one name of the concept might come from its inventor, whereas another name is lexically descriptive (e.g., "Karoubi envelope" is synonymous to the type "category of idempotent arrows"). Third, concepts in mathematics can be abstracted using constructs from various mathematical frameworks. Often, a correspondence between concepts can be formed across different theories, each with its own palette of types. For example, a "magma" in group theory is a generalisation of a "groupoid", which can be defined as a special kind of "category" in category theory. Mathematicians have the flexibility to map mathematical concepts between theories, perform reasoning in one theory and then project back to another theory.

Methods that exploit semantic type relatedness, such as TypesExp, can be advantageous in cases where (a) lexical sparsity requires the use of some form of generalisation or similarity across concepts, of which QE is a simple variant, and (b) information about the similarity between types is more informative than similarity between raw words.

6 Conclusions

We have shown experimentally that types are a valuable, but currently underutilised aspect of the mathematical linguistic discourse. Our model improves MIR of research-level queries by automatically identifying types in text and building a similarity (embedding) space with which related types can be detected in documents more effectively than with existing methods. We find that type-based query expansion using this method of semantic proximity outperforms state-of-the-art IR/expansion models on a realistic, large-scale test collection. This strongly suggests that it is the identification of semantic relationships between types that can improve the quality of research-level MIR in the future even further. As an additional advantage, prior knowledge of types may also help improve parts of an NLP pipeline for mathematics texts, particularly in the absence of domain-specific training material.

References

Nasreen Abdul-jaleel, James Allan, W. Bruce Croft, O Diaz, Leah Larkey, Xiaoyan Li, Mark D. Smucker, and Courtney Wade. 2004. Umass at trec 2004: Novelty and hard. In *In Proceedings of TREC-13*.

Akiko Aizawa, Michael Kohlhase, and Iadh Ounis. 2013. Ntcir-10 math pilot task overview. In *Proceedings of the 10th NTCIR Conference*, June.

Akiko Aizawa, Michael Kohlhase, Iadh Ounis, and Moritz Schubotz. 2014. NTCIR-11 math-2 task overview. In *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014*.

Ronan Cummins, Jiaul H. Paik, and Yuanhua Lv. 2015. A pÓlya urn document language model for improved information retrieval. *ACM Trans. Inf. Syst.*, 33(4):21:1–21:34, May.

Joseph L. Fleiss. 1971. Measuring nominal scale agreement among many rater. *Psychological Bulletin*, 76:378–382.

Katerina T. Frantzi, Sophia Ananiadou, and Jun-ichi Tsujii. 1998. The c-value/nc-value method of automatic recognition for multi-word terms. In *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, ECDL '98, pages 585–604, London, UK, UK. Springer-Verlag.

Mohan Ganesalingam. 2008. *The Language of Mathematics*. Ph.D. thesis, Cambridge University Computer Laboratory.

Mihai Grigore, Magdalena Wolska, and Michael Kohlhase. 2009. Towards context-based disambiguation of mathematical expressions. In *The joint conference of ASCM 2009 and MACIS 2009. 9th international conference on Asian symposium on computer mathematics and 3rd international conference on mathematical aspects of computer and information sciences, Fukuoka, Japan, December 14–17, 2009. Selected papers.*, pages 262–271. Fukuoka: Kyushu University, Faculty of Mathematics.

Ferruccio Guidi and Claudio Sacerdoti Coen. 2015. A survey on retrieval of mathematical knowledge. *CoRR*, abs/1505.06646.

Radu Hambasan, Michael Kohlhase, and Corneliu-Claudiu Prodescu. 2014. Mathwebsearch at NTCIR-11. In *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014*.

Stephen P. Harter. 1975. A probabilistic approach to automatic keyword indexing. part i. on the distribution of specialty words in a technical literature. *Journal of the American Society for Information Science*, 26(4):197–206.

John S. Justeson and Slava M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Michael Kohlhase and Corneliu-Claudiu Prodescu. 2013. Mathwebsearch at NTCIR-10. In *Proceedings of the 10th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-10, National Center of Sciences, Tokyo, Japan, June 18-21, 2013*.

Giovanni Yoko Kristianto, Minh quoc Nghiem, Yuichiroh Matsubayashi, and Akiko Aizawa. 2012. Extracting definitions of mathematical expressions in scientific papers. In *In JSAI*.

Giovanni Yoko Kristianto, Goran Topic, and Akiko Aizawa. 2014. Exploiting textual descriptions and dependency graph for searching mathematical expressions in scientific papers.

Martin Líška, Petr Sojka, Michal Růžička, and Petr Mravec. 2011. Web interface and collection for mathematical retrieval: Webmias and mrec. In Petr Sojka and Thierry Bouche, editors, *Towards a Digital Mathematics Library.*, pages 77–84, Bertinoro, Italy, Jul. Masaryk University.

Yuanhua Lv and ChengXiang Zhai. 2009. A comparative study of methods for estimating query language models with pseudo feedback. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 1895–1898, New York, NY, USA. ACM.

Yuanhua Lv and ChengXiang Zhai. 2011. Lower-bounding term frequency normalization. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, pages 7–16, New York, NY, USA. ACM.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Nidhin Pattaniyil and Richard Zanibbi. 2014. Combining TF-IDF text retrieval with an inverted index over symbol pairs in math expressions: The tangent math search engine at NTCIR 2014. In *Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014*.

Minh Nghiem Quoc, Keisuke Yokoi, Yuichiroh Matsubayashi, and Akiko Aizawa. 2010. Mining coreference relations between formulas and text using wikipedia.

S. E. Robertson and S. Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '94, pages 232–241, New York, NY, USA. Springer-Verlag New York, Inc.

S. E. Robertson, C. J. van Rijsbergen, and M. F. Porter. 1981. Probabilistic models of indexing and searching. In *Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval*, SIGIR '80, pages 35–56, Kent, UK. Butterworth & Co. #38.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at trec-3. In Donna K. Harman, editor, *TREC*, volume Special Publication 500-225, pages 109–126. National Institute of Standards and Technology (NIST).

Mark D. Smucker, James Allan, and Ben Carterette. 2007. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 623–632, New York, NY, USA. ACM.

Petr Sojka and Martin Líška. 2011. The art of mathematics retrieval. In Frank Wm. Tompa Matthew R. B. Hardy, editor, *Proceedings of the 2011 ACM Symposium on Document Engineering*, pages 57–60, Mountain View, CA, USA. ACM.

Yiannos Stathopoulos and Simone Teufel. 2015. Retrieval of research-level mathematical information needs: A test collection and technical terminology experiment. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 334–340.

Ellen M. Voorhees and Donna K. Harman. 2005. *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press.

Chengxiang Zhai and John Lafferty. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 334–342, New York, NY, USA. ACM.

Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, April.