Corpus-based Population of a Mid-level Business Ontology

András Kornai

MTA SZTAKI

Abstract. We describe the creation of a broad mid-level ontology, several thousand nodes, suitable for classification and analysis of business documents of the kind regularly kept in corporate document storage. The main claim of the paper is that we can populate a rich mid-level ontology by largely automatic, corpus-based methods.

1 Introduction

In Section 2 we begin by reviewing some standard notions, and describe the principles of what we will refer to as *Midlevel Business Ontology* (MBO). These principles guide the learning process that is used to extract an actual ontology of over 5k entities from a corpus of 20k documents of the kind found in corporate document storage: memos, activity plans, agendas, proposals, CVs, regulatory (legal) documents, accounting materials, bills, invoices, letters (including emails), etc. In Section 3 we describe the process of node selection, and in Section 4 we describe the data cleaning process. We believe our chief method, the iterative sharpening of linear classifiers, is also applicable to the problem of automatically building a rudimentary hierarchy among the entries, and we conclude the paper with some programmatic remarks on this.

2 Linking MBO to well-known upper ontologies

We assume, without argumentation, the standard tripartite division into high-, mid- and low-level ontologies. For the high level (also known as *upper, top,* or *foundational*) ontology, we use the 4lang ontology [1] now better called 40lang, inasmuch as bindings have been extended to 40 languages [2].

Perhaps the major division line among various ontologies is whether they are intended to capture knowledge about the world (e.g. about distinctions among various physical objects such as tools) or about conceptual entities. To put this another way, we must decide whether it is the difference between *hammers* and *nails* that we are intent on systematizing, or the difference between the concepts (words, mental/cognitive entities) 'hammer' and 'nail'. Since our interest is with the latter, our work is more closely related to ontologies like DOLCE than to word taxonomies like WordNet [3].

The very same object, say an MS Word file preserved on a particular floppy disk, can be a 'contract', a 'draft', or an 'exhibit', which means that very different rules apply to it – drafts can be modified at will, while tampering with evidence is a crime. At the same time, different objects, such as the file as it appears on the hard drive, in hardcopy, or in an email attachment preserved on a computer on another continent, may relevantly be called the same.

Cataloging physical objects remains a valid goal for ontologies, but to use MBO for this purpose it would have to be supplemented by some system of physical or logical coordinates which lies outside the business ontology proper. The main lesson we take away from physical objects is that none of them are true endurants: things have a beginning (creation process) and end (destruction process). This is evident for business objects like contracts or offers, but in MBO we treat more enduring abstract objects like laws and regulations the same way.

For a full ontology, we would need three kinds of information: pure generic, modified generic, and domain-specific. By *pure generic* information we mean the kind of very general statement that objects (typically, nouns) can be divided in two basic classes, 'physical' and 'conceptual', with mass, energy, and space-time coordinates easily attached to the former, but not the latter, while requirements, obligations, etc. are easily attached to the latter but not the former. Statements at this high level of generality apply within the business domain just as well as in any other domain we can think of, such as the medical or the legal domains, and thus belong in the top-level ontology.

With a thousand or so entries, 4lang is considerably richer than the philosophically inspired top-level ontologies, and contains many words that we call *modified generics*. For an example, consider *charge*, which is in a business context tied to fees 'merces', in a legal context to 'accusatio', both of which modify the general meaning of charge as some kind of attack 'impetus' quite substantially. The overlap between 4lang and the raw mid-level list is a rich source of examples of this phenomenon, but we find even more examples among words that are not considered basic and are thus not present in 4lang: consider for example the verb *to hedge*. Outside the business domain, this means 'to avoid giving a promise or direct answer' (Merriam-Webster), within the domain the prevalent meaning is 'to buy or sell commodity futures as a protection against loss due to price fluctuation'. The two meanings share the common element 'to evade the risk of commitment' but the technical means of carrying out the evasion are very different.

Finally, for an example of a *domain-specific* concept consider *budget* (both noun and verb). It is possible to use this word in another domain, e.g. a newspaper story about a boxing match may say that the loser didn't budget enough energy for the final round, but by doing so the writer invokes the business metaphor (a reversal of the more common 'business is war' trope). To capture the truly business-specific vocabulary we need to proceed top down, building out some core scripts, such as `retail business`, where products are sold to customers, `rental business`, where products are leased or rented to customers, `service, market`, and so forth. All these core scripts have the same typecasting power

over their components: we may normally think of surgical wards in the medical context, where an appendictomy is 'an urgent life-saving procedure', but we may also think of them as retail stores, where appendictomies are products, sold to customers. These customers happen to be called 'patients', but business is business, we first need to establish their capacity to pay.

Since our overarching goal is to establish the business-specific concepts, including the business-specific readings of generic concepts, with as little human intervention as possible, we need to divide the corpus into documents that are, ideally, reflective only of a single core category such as `retail`. For this we need to clean the corpus of material that belongs, according to human judgment, to two or more (sub)domains at the same type: typical examples would be a document that describes the procedure for testing financial software, as it belongs both to `banking` and `IT`, or plans for customer-facing operations (`retail`) for an organization that normally operates upstream (`wholesale`).

3 Automatic acquisition of nodes

Part of the challenge in building mid-level ontologies comes from above, linking with the top level, and part comes from below, in trying to link with specific low-level ontologies and knowledge bases. But there are two challenges that are intrinsic to the middle level: populating the domain ontology, and keeping it free from material from other mid- or low-level domains. One way to build MBO would be based on introspection, but it is hard to find experienced businessmen who are also sophisticated lexicographers, ontology builders, and knowledge engineers at the same time. Here we describe how we can select a rich mid-level ontology based on a corpus, and defer the issue of keeping out polluting material to Section 3.

We begin with the CS corpus of about 20k documents selected randomly from the servers of a well known multinational firm (over EUR 10 billion in annual revenues and over 100k employees) that offers professional services to other businesses, guaranteeing that the vocabulary extracted from it is not restricted to any vertical. (As it currently stands, the CS corpus is not available to the public, but efforts to suitably anonymize it are under way.) The 27m word tokens are in 453k types, of which 216,450 occur more than once (hapax legomena are discarded). The rest was compared to the Google 1T vocabulary [4] in several steps.

First, we considered the words unique to the CS corpus and order these by frequency. At the top we only find expressions such as *N/A* or *follow-up* which are missing from G1T only because Google is using a different tokenization algorithm, which splits on slash and hyphen. In fact, over 70% of the 103k words that do not appear in G1T are the result of such mismatches, and the remainder is dominated by token classes whose individual words are of little interest, such as numbers like *56101363*; SQL and other programming language keywords such as *VARCHAR25*; and table column headers like *StateIncluded*. Once these Information Technology (IT) words are discarded, by a data cleaning

process we defer to Section 3, we are left with only 1,722 words, the majority of which are foreign, typically French, Italian, German, and Spanish, reflecting the international nature of business. If these are removed and typos are discarded, we are left with only 85 words (in order of decreasing frequency):

subinventories, preadmit, autocash, promptable, billdate, tradelane, termdate, userviews, coverdoc, workrequest, substatuses, ratecode, preadmitted, megaprocesses, finaldoc, callbase, autoinvoicing, autoaccounting, acceptancetest, totalcharges, totalbarrels, recruitability, minispecs, invoiceless, soustotals, salesorders, workstructure, videocypher, sidemarking, preadmits, modelclass, modelcategory, desginator, wellnumber, prebonus, blueplate, waybilling, subnetworked, subledgering, subinstitutional, strawmans, stocknumbers, recoupability, rebillable, reapproves, prebilled, postbilling, outcomedoc, multifacilities, memodoc, intraoperation, hitchment, budgetxls, btuvalue, unissue, shipvendor, shiftwise, sheetmetals, settlementdoc, settlebatch, screenpainter, saleorder, retrieveability, reputs, reportxls, reportsdoc, palettization, nonclearable, newquantity, matrixtesting, matrixdoc, matricesdoc, materialsql, masterdoc, manweeks, knowledgeweb, jobchangeover, inputdoc, guidelinesdoc, detailable, dealsheets, bundletracker, autosourcing, autoscheduled.

Many of these are either whitespace errors or, more likely, also column headings: *term date, work request, settlement doc,* etc. With a high quality morphological analyzer we can find many others that appear in G1T in their citation (stem) form: *subinventory, substatus, preadmit,* etc., and once these are taken into account, we are left with a handful of compounds and latinate formations (particularly prefixes *pre-, sub-, un-, re-, intra-,* see [5]) that are truly characteristic of business vocabulary. Overall, words that are missing from G1T are not a significant source of MBO candidates.

Next we consider those 103k words that appear both in CS with absolute frequency $F > 1$ and in G1T with absolute frequency $G > 100$ (the cutoff of the Google count). We exclude proper names (since the corpus is not yet anonymized), which reduces the corpus to 30k word forms. Since the G1T corpus is much larger (by a factor of about 25,000), $\log(G/F)$ is on the average 9.93, with a variance of 2.22. Therefore, it makes sense to restrict attention to those words where this number is below average, i.e. those words that are used *at least as often* in business documents as in general English. Only 14k word forms meet this criterion, and a quarter of these are foreign. We can remove the bulk of these by prefiltering the corpus for language.

Of the remaining 10,227 words we consider only those 6,039 that appear at least in 9 documents. The publicly available mid-level entity list (for which see `http://hlt.sztaki.hu/resources`) is cleaned of typos (including proper names that were left uncapitalized) but not fully stemmed. Since this is a departure from standard lexicographic practice, let us briefly describe the reason for keeping non-stemmed (often derived, but sometimes even inflected) forms. Consider, for example, the adjective *yearly*, obtained from the stem *year* by an entirely regular, highly productive suffix. Since there is nothing business-specific about the word *year* or the way this word is used in business documents, it clearly doesn't belong

in MBO. But in the business context *yearly* carries a sense of obligation that is missing from the generic use – iceberg formation or stork migration happen yearly, but are not obligations. This is quite consistent with the fact that the relative frequency of *year* is the same in the business domain as in English in general, while the frequency of *yearly* is almost twice (1.92 times) as large.

In fact, higher than expected proportion of derived forms is a good predictor of domain-specificity. Consider a plural like *customizations* or a past tense like *architected* – these are far less likely in environments where *customization* is not a frequent noun and *to architect* is not a frequent verb to begin with. Though random spot-checks of material from other domains bear out the validity of this observation, we have something of a chicken-and-egg problem here, in that we cannot at the same time claim that our material proves the observation and use the observation to select the material. In this paper, we chose to take the validity of our observation on faith, and use it instrumentally to select the MBO nodes – independent verification must await the public availability of domain-specific corpora and their independently arrived at mid-level ontologies.

4 Cleaning the data

Ideally, we would want to begin with a few well understood scripts (in the sense of Schank and Abelson [6]) such as 'selling', 'investing', and other prominent business activities, but this would again lead us to the problem that we started out with, that there are very few domain experts who are also knowledge engineers. Thus we seek a less perfect automated or semi-automated solution, one that clusters the mid-level data in script- or frame-sized subdomains, ideally with minimal overlap. Of course eliminating overlap cannot be taken to the extreme: every business operates in some domain, often more than one, and if we omitted every `retail` document that is about `apparel` or `automotive` or similar verticals we'd be left with nothing.

Manual inspection of the raw entity list made clear that we have a significant number of documents containing terms that are highly specific to information technology (`IT`): not just programming terms like *alloc, atoi, fflush, fprintf, ...* but also expressions associated to the high-level planning stages such as *alphanum, autoexec, flowchart, gigabyte, groupware,* etc. Here we had to make a strategic decision, whether to treat IT as yet another business domain, or segregate the IT-specific vocabulary. Since our data was obtained via IT consulting, in the interest of a balanced ontology we chose the latter method, but we emphasize that the algorithm used for doing so is just as applicable to the `IT` versus `non-IT` decision as it is to `retail` versus `wholesale`.

In stage 0, we begin with a manually selected seed list of IT-specific words such as the ones listed above, and observe their probability in the corpus. We compute a simple but effective linear classifier (see [7]) that uses the *relevance* (defined as the difference between the log frequency in the positive set and the log frequency in the background model) of keywords and key phrases for weights, retaining only those keywords/phrases whose relevance exceeds some *threshold*

of significance τ, say $\tau = 3$. At this stage, we use the G1T count for background. The stage 0 classifier is thus a simple relevance-weighted word vector, which is multiplied with the TF vector of each document to obtain a *raw score* that gets normalized by dividing it by $n^{0.8}$, where n is the number of words in the document. (Here 0.8 is the Herdan-Heaps exponent, see [8] and [9]).

In stage 1, we rank the documents by the stage 0 classifier, and cut off the list by manual inspection so as to include only evidently IT-specific documents. Techniques for automating this step are of great interest, but would take us far from our immediate goals of populating the ontology and building the knowledge base. We now repeat the frequency count on the selected documents, and rerank the words, using either G1T or the overall corpus frequencies as background for establishing the relevances. The process can be iterated as many times as we wish, limited only by our ability to cut off the document lists (which is easy by binary search). A list of some 80 highly IT-specific terms obtained this way is included here:

abend abends alphanum autocreate autofill configurator customization customizations datafield datafiles datawindow datawindows dbase dbms deliverables dialer downtime esc fileserver flowchart flowsheet flowsheets fprintf functionality indirects inputters intercompany interfaced jobcode jpl keytab mainframes maint masterfiles matchcode matchcodes matl middleware mmddyy mmddyyyy parm parms pcs procs pseudocode redisplay redisplayed redisplays reformats routings rowid rqmt runscript signoff signoffs signon spoolfile sprintf sqlplus sqr strcat strcpy strncat strncmp strncpy strupr submenu subprocesses subsystem sybase systime tabbing tableset tablespace timestamp tinyint toupper userview varchar.
As we discussed in Section 2, such lists are likely to contain many terms like *redisplay redisplayed redisplays* that stemming would collapse in a single term, but this would not be desirable in that domain-specific terms like *indirects* would by such a process be reduced to terms like *indirect* that are no longer specific to the domain. Practical experience with these classifiers shows that removing all but the top d keys ($20 \leq d \leq 200$) by aggressive thresholding decreases the recall of the classifiers by very little and their precision even less, and that the key issue driving performance is the choice of words/phrases kept rather than their exact weight.

The algorithm is best analyzed in the frame of PAC-learning [10]. Our *sample space* S is the corpus, our *concept* C to learn was IT above, but could be any other mid-level concept like `insurance`. We are interested in learning the concept with $1-\delta$ probability and ϵ precision, with δ, ϵ in the 1-10% range, which is practically feasible, even though the theoretical bound based on VC dimension ($d+1$ for a linear classifier) falls short of what we want for this size ($N \approx 20,000$) data set.

5 Conclusions, future directions

The main claim of this paper was that from a raw corpus of some 20k business documents we can populate a sizeable mid-level ontology with minimal human intervention. While we cannot at present make the corpus publicly accessible

(anonymization is still under way), we make the the raw concept list of 5,779 entries downloadable from `http://hlt.sztaki.hu/resources`.

This list, for the reasons discussed in the paper, is still a mixture of morphologically complex (derived, inflected) and morphologically simplex (stem) forms. By automatic stemming we would lose both ontological insight (e.g. that *versioning* is not just the process of making versions) and discriminative power in classification tasks. Once the hierarchization is complete, we expect the list to shrink to half of its current size, still quite large for a mid-level ontology.

The next steps are building the hierarchy and attaching definitions to each concept. Our plan is to generalize PAC concept learning to the case of learning several concepts together. Broadly speaking, instead of a single linear classifier and the attached document set we try to bootstrap k classifiers such that the associated k document sets are largely disjoint and exhaustive. For each domain we start with small, manually created seed lists e.g. for `retail` we would have *customer discount price purchase retail seller store*, for `banking` we would have *account, atm, cd, checking, loan, savings* and so forth, for a few dozen subdomains.

In each iteration, we cluster the documents, and investigate how well the classifiers capture these. For now, we have now way of automating this manual supervision step, but we note that such spotchecks require a great deal less labor than manually classifying the entire corpus to different subdomains.

Acknowledgment

We thank Judit Ács and Attila Zséder for their help at various stages of the pipeline.

References

1. Kornai, A., Makrai, M.: A 4lang fogalmi szótár [The 4lang concept dictionary]. In Tanács, A., Vincze, V., eds.: IX. Magyar Számitógépes Nyelvészeti Konferencia [Ninth Conference on Hungarian Computational Linguistics]. (2013) 62–70
2. Ács, J., Pajkossy, K., Kornai, A.: Building basic vocabulary across 40 languages. In: Proceedings of the Sixth Workshop on Building and Using Comparable Corpora, Sofia, Bulgaria, Association for Computational Linguistics (2013) 52–58
3. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A.: Sweetening WordNet with DOLCE. AI Magazine **24**(3) (2003) 13–24
4. Brants, T., Franz, A.: Web 1T 5-gram Version 1. Linguistic Data Consortium, Philadelphia (2006)
5. Aronoff, M.: Word Formation in Generative Grammar. MIT Press (1976)
6. Schank, R.C., Abelson, R.P.: Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures. Lawrence Erlbaum, Hillsdale, NJ (1977)
7. Kornai, A., Krellenstein, M., Mulligan, M., Twomey, D., Veress, F., Wysoker, A.: Classifying the Hungarian Web. In Copestake, A., Hajic, J., eds.: Proceedings of the EACL. (2003) 203–210
8. Herdan, G.: Quantitative linguistics. Butterworths, Washington (1964)

9. Heaps, H.S.: Information Retrieval – Computational and Theoretical Aspects. Academic Press (1978)
10. Valiant, L.G.: A theory of the learnable. Communications of the ACM **27**(11) (1984) 1134–1142