First Experiments with Data Driven Conjecturing*

Karel Chvalovský[1], Thibault Gauthier[1], and Josef Urban[1]

Czech Technical University in Prague, Czech republic,
karel@chvalovsky.cz, email@thibaultgauthier.fr , and josef.urban@gmail.com

An essential part of mathematics and the work of a mathematician is to produce conjectures. This is also an important problem in automated theorem proving. (Un)fortunately, already humans have different opinions on what is a good conjecture and hence an objective function for ranking conjectures is hard to specify. It is even less clear how conjectures are discovered.

There have been various attempts to produce conjectures automatically. Well known examples like Lenat's AM (Automated Mathematician) [9], a more specialized Graffitti by Fajtlowicz [4], and Colton's HR [3] are based on human curated rules for generating conjectures. In small domains, exhaustive brute force generation can be useful, in particular when controlled by a type system and further semantic pruning [7].

Using Distributed Representations for Conjecturing: Our approach is different. We do not want to write down rules describing interesting conjectures directly, but we would like to *learn* meaningful conjecturing from a large corpus of mathematical proofs. For that, a better semantic understanding of such corpora is needed. It is possible to use distributional semantics approach, where we try to learn semantic similarities among concepts solely based on their co-occurrences in a corpus. This has proven to be very successful in computational linguistics [10]. A *notion* (concept) is then represented by a low-dimensional vector. One of the interesting aspects of such a representation are analogies via linear algebra. Let $\mathbf{v}_\cap$, $\mathbf{v}_\cup$, and $\mathbf{v}_\wedge$ be the vector representations of $\cap$, $\cup$, and $\wedge$, respectively. Then we can answer a question *"What is to $\wedge$ as $\cup$ is to $\cap$?"* by finding $\mathbf{v}$ such that $\mathbf{v}_\wedge - \mathbf{v}$ is most similar to $\mathbf{v}_\cap - \mathbf{v}_\cup$. Such analogies can be used for free-style conjecturing similar to [6].

A straightforward application of this idea is to learn such representations over a large formal library, in our case we use the Mizar [2] Mathematical Library (MML). Given a statement s, for example, $x \cap y = x \to x \cup y = y$, we can identify an important notion in s that we would like to shift, e.g., $\cap$ represented by $\mathbf{v}_\cap$. Now we look for a vector that is close to $\mathbf{v}_\cap$ such that it is a binary function. If we are lucky $\mathbf{v}_\wedge$ is close and hence we would like to replace $\cap$ in s by $\wedge$. We should also replace $\cup$ by a binary function represented by a vector that is to $\mathbf{v}_\cup$ as $\mathbf{v}_\wedge$ is to $\mathbf{v}_\cap$. It could be $\mathbf{v}_\vee$ and hence we obtain $x \wedge y = x \to x \vee y = y$ as a new statement.

However, here we have made several decisions and it is rather unclear how to make them automatically. Before we start to discuss them, it is worth mentioning that our situation is significantly different from the situation in natural language processing (NLP). We use the Mizar formal library so for every statement we have a parse tree. Moreover, if we produce a new statement from an old one, we can try to check by an automated theorem prover (ATP) whether it is provable or disprovable, because we can work directly with a TPTP [11] translation of the Mizar statement [12]. Although it is generally very difficult to disprove a statement, in our case it is possible to do that for trivially invalid statements, which we will often produce. Similarly, we can filter trivially valid statements.

Now back to our problem. We can use a distributed representation of notions and statements such as [1]. Given a statement s we can find an important notion N in it and shift it (i.e., its vector). Here N should be a predicate, function, or a constant. Once we do that, we can

*Supported by the ERC Consolidator grant no. 649043 AI4REASON and by the Czech project AI&Reasoning CZ.02.1.01/0.0/0.0/15_003/0000466 and the European Regional Development Fund.

look for a semantically similar notion of the same type. This search can be unrestricted, or we can look for notions that, e.g., appear only in different Mizar articles. When we find a suitable notion (or more notions), we can start to shift notions in our statement from the most important to the least important. It is unlikely that a vector for a new notion is exactly at the position where we expect it, therefore we can use the previous shifts to correct the new ones. The question when to stop shifting and keep the rest of the statement intact can be left open, because we can generate all possible variants and remove those that are trivially (in)valid.

This procedure can be improved in various ways, e.g., by using a beam search with the possibility to keep a notion intact even when less important notions are shifted. However, so far we have obtained only weak results with this method. It probably suffers from the fact that it is hard to keep all parts synchronized. A single error can spoil the whole translation, and even more importantly, it is usually necessary to shift different parts of statements differently.

Consistency by NMT: The recently developed neural machine translation (NMT) architectures provide a different and possibly better approach. It was shown recently that we can use NMT for simple informal to formal translations [14]. Here, the above mentioned semantic relations between the notions are learned as part of the training process. Moreover, the inner consistency of the translated result is controlled directly by NMT. That is even incorrect results are likely to parse and the notions are combined meaningfully. For example, in the encoder-decoder neural architectures a hidden state (vector) characterizing the translation done so far is updated after each decoding step, and the choice of the next decoded symbol is statistically conditioned on the state, making the resulting combinations of symbols statistically plausible.

We can formulate our conjecturing task as a translation problem—translate an already known statement s into a conjecture t. How can we produce a sufficient amount of training data $\{\,(s,t)\colon s$ translates into $t\,\}$ for such a task? Assume we have a statement s and we can say that statements $t_1,\ldots,t_n$ in our library are somehow relevant to s. We can then try to confuse NMT by adding n training examples $\{(s,t_1),\ldots,(s,t_n)\}$ and hence NMT will then attempt to translate s into a statement that is most similar to all $t_1,\ldots,t_n$.

For an initial experiment, we produce abstracted common patterns (e.g. commutativity, associativity, etc.) from all Mizar toplevel statements using Gauthier's patternizer [5] used previously for concept alignment and conjecturing based on them [6]. The patternizer finds about 16000 patterns that generalize at least two statements. From them we create a corpus of about 1.3 million (non-unique) translation pairs by making an input-output pair from all statements that are instances of the same pattern. This means that NMT will be trained to analogize on many examples, and due to the large non-determinism in the training data it may produce a new formula that will likely be syntactically consistent. This is indeed often the case on a test set of about 30000 unique statements that after the training result in about 16000 formulas that do not appear in MML. A very simple example generated by this conjecturing approach is $(X\cap Y)\setminus Z=(X\setminus Z)\cap(Y\setminus Z)$ produced from $(X\cup Y)\setminus Z=(X\setminus Z)\cup(Y\setminus Z)$.

Although it is a trivial duality statement, it should be noted that it was produced completely automatically without any intervention from outside and it is not in the Mizar library. Moreover, there is no need to check for a correct substitution, cf. [5], this part is handled by NMT itself. This statement can be proved automatically by the MizAR hammer [13, 8]. Examples of false but syntactically consistent conjectures generated automatically in this way include:

```
for n, m being natural numbers holds n gcd m = n div m;
for R being Relation holds with_suprema(A) <=> with_suprema(inverse_relation(A));
```

In this initial experiment, we say that two statements with a common pattern are relevant to each other. There are many other untested options, for example, we can say that a statement t is relevant to s if t occurs in a proof of s, or vice versa.

References

[1] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR 2017*, 2017.

[2] Grzegorz Bancerek, Czeslaw Bylinski, Adam Grabowski, Artur Kornilowicz, Roman Matuszewski, Adam Naumowicz, Karol Pak, and Josef Urban. Mizar: State-of-the-art and beyond. In Manfred Kerber, Jacques Carette, Cezary Kaliszyk, Florian Rabe, and Volker Sorge, editors, *Intelligent Computer Mathematics - International Conference, CICM 2015, Washington, DC, USA, July 13-17, 2015, Proceedings*, volume 9150 of *Lecture Notes in Computer Science*, pages 261–279. Springer, 2015.

[3] Simon Colton. *Automated Theory Formation in Pure Mathematics*. Distinguished Dissertations. Springer London, 2012.

[4] Siemion Fajtlowicz. On conjectures of Graffiti. *Annals of Discrete Mathematics*, 72(1–3):113–118, 1988.

[5] Thibault Gauthier and Cezary Kaliszyk. Aligning concepts across proof assistant libraries. *J. Symb. Comput.*, 90:89–123, 2019.

[6] Thibault Gauthier, Cezary Kaliszyk, and Josef Urban. Initial experiments with statistical conjecturing over large formal corpora. In Andrea Kohlhase, Paul Libbrecht, Bruce R. Miller, Adam Naumowicz, Walther Neuper, Pedro Quaresma, Frank Wm. Tompa, and Martin Suda, editors, *Joint Proceedings of the FM4M, MathUI, and ThEdu Workshops, Doctoral Program, and Work in Progress at the Conference on Intelligent Computer Mathematics 2016 co-located with the 9th Conference on Intelligent Computer Mathematics (CICM 2016), Bialystok, Poland, July 25-29, 2016.*, volume 1785 of *CEUR Workshop Proceedings*, pages 219–228. CEUR-WS.org, 2016.

[7] Moa Johansson, Dan Rosén, Nicholas Smallbone, and Koen Claessen. Hipster: Integrating theory exploration in a proof assistant. In Stephen M. Watt, James H. Davenport, Alan P. Sexton, Petr Sojka, and Josef Urban, editors, *Intelligent Computer Mathematics - International Conference, CICM 2014, Coimbra, Portugal, July 7-11, 2014. Proceedings*, volume 8543 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2014.

[8] Cezary Kaliszyk and Josef Urban. MizAR 40 for Mizar 40. *J. Autom. Reasoning*, 55(3):245–256, 2015.

[9] Douglas Bruce Lenat. *AM: An Artificial Intelligence Approach to Discovery in Mathematics as Heuristic Search*. PhD thesis, Stanford, 1976.

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119, 2013.

[11] Geoff Sutcliffe. The TPTP problem library and associated infrastructure. *J. Autom. Reasoning*, 43(4):337–362, 2009.

[12] Josef Urban. MPTP 0.2: Design, implementation, and initial experiments. *J. Autom. Reasoning*, 37(1-2):21–43, 2006.

[13] Josef Urban, Piotr Rudnicki, and Geoff Sutcliffe. ATP and presentation service for Mizar formalizations. *J. Autom. Reasoning*, 50:229–241, 2013.

[14] Qingxiang Wang, Cezary Kaliszyk, and Josef Urban. First experiments with neural translation of informal to formal mathematics. In Florian Rabe, William M. Farmer, Grant O. Passmore, and Abdou Youssef, editors, *Intelligent Computer Mathematics - 11th International Conference, CICM 2018, Hagenberg, Austria, August 13-17, 2018, Proceedings*, volume 11006 of *Lecture Notes in Computer Science*, pages 255–270. Springer, 2018.