Private Access to Phrase Tables for Statistical Machine Translation

Nicola Cancedda
Xerox Research Centre Europe
6, chemin de Maupertuis
38240, Meylan, France
Nicola.Cancedda@xrce.xerox.com

Abstract

Some Statistical Machine Translation systems never see the light because the owner of the appropriate training data cannot release them, and the potential user of the system cannot disclose what should be translated. We propose a simple and practical encryption-based method addressing this barrier.

1 Introduction

It is generally taken for granted that whoever is deploying a Statistical Machine Translation (SMT) system has unrestricted rights to access and use the parallel data required for its training. This is not always the case. The ideal resources for training SMT models are Translation Memories (TM), especially when they are large, well maintained, coherent in genre and topic and aligned with the application of interest. Such TMs are cherished as valuable assets by their owners, who rarely accept to give away wholesale rights to their use. At the same time, the prospective user of the SMT system that could be derived from such TM might be subject to confidentiality constraints on the text stream needing translation, so that sending out text to translate to an SMT system deployed by the owner of the PT is not an option.

We propose an encryption-based method that addresses such conflicting constraints. In this method, the owner of the TM generates a Phrase Table (PT) from it, and makes it accessible to the user following a special procedure. An SMT decoder is deployed by the user, with all the required resources to operate except the PT[1].

As a result of following the proposed procedure:

- The user acquires all and only the phrase table entries required to perform the decoding of a specific file, thus avoiding complete transfer of the TM to the user;
- The owner of the PT does not learn anything about what is being translated, thus satisfying the user's confidentiality constraints;
- The owner of the PT can track the number of phrase-table entries that was downloaded by the user.

The method assumes that, besides the PT *Owner* and the PT *User*, there is a *Trusted Third Party*. This means that both the User and the PT owner trust such third party not to collude with the other one for violating their secrets (i.e. the content of the PT, or a string requiring translation), even if they do not trust her enough to directly disclose such secrets to her.

While the exposition will focus on phrase tables, there is nothing in the method precluding its use with other resources, provided that they can be represented as look-up tables, a very mild constraint. Provided speed-related aspects can be dealt with, this makes the method directly applicable to language models, or distortion tables for models with lexicalized distortion (Al-Onaizan and Papineni, 2006). The method is also directly applicable to Translation Memories, which can be seen as "degenerate"

[1] If the decoder can operate with multiple PTs, then there could be other (possibly out-of-domain) PTs installed locally.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 23–27,
Jeju, Republic of Korea, 8-14 July 2012.

phrase tables where each record contains only a translation in the target language, and no associated statistics.

The rest of this paper is organized as follows: Section 2 explains the proposed method; in Section 3 we make more precise some implementation choices. We briefly touch on related work on Section 4, provide an experimental validation in Sec. 5, and offer some concluding remarks in Sec. 6.

2 Private access to phrase tables

Let Alice[2] be the owner of a PT, Bob the owner of the SMT decoder who would like to use the table, and Tina a trusted third-party. In broad terms, the proposed method works like this: in an initialization phase, Alice first encrypts PT entries one by one, sends the encrypted PT to Bob, and the encryption/decryption keys to Tina. Alice also sends a method to map source language phrases to PT indices to Bob.

When translating, Bob uses the mapping method sent by Alice to check if a given source phrase is present and has a translation in the PT and, if this is the case, retrieves the index of the corresponding entry in the PT. If the check is positive, then Bob sends a request to Tina for the corresponding decryption key. Tina delivers the decryption key to Bob and communicates that a download has taken place to Alice, who can then increase a download counter.

Let $\{(s_1, v_1), \ldots, (s_n, v_n)\}$ be a PT, where s_i is a source phrase and v_i is the corresponding record. In an actual PT there are multiple lines for a same source phrase, but it is always possible to reconstruct a single record by concatenating all such lines.

2.1 Initialization

The initialization phase is illustrated in Fig. 1. For each PT entry (s_i, v_i), Alice:

1. Encrypts v_i with key k_i We denote the encrypted record as $v_i \oplus k_i$
2. Computes a *digest* d_i of the source entry s_i
3. Sends the phrase digests $\{d_i\}_{i=1,\ldots,n}$ to Bob

[2]We adopt a widespread convention in cryptography and assign person names to the parties involved in the exchange.

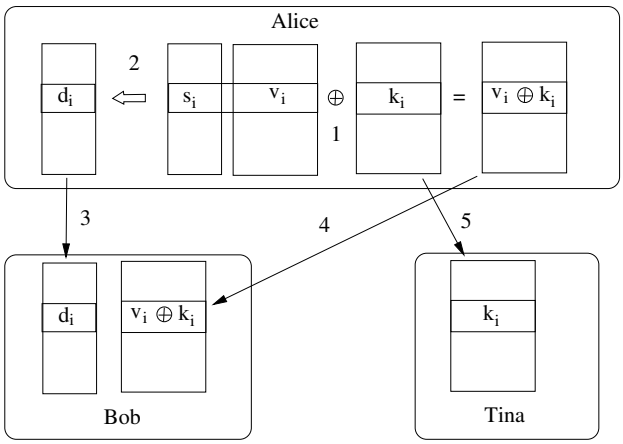

Figure 1: The initialization phase of the method (Sec. 2.1). Bob receives an encrypted version of the PT entries and the corresponding source phrase digests. Tina receives the decryption keys.

4. Sends the encrypted record (or *ciphertext*) $\{v_i \oplus k_i\}_{i=1,\ldots,n}$ to Bob
5. Sends the keys $\{k_i\}_{i=1,\ldots,n}$ to Tina

A *digest*, or *one-way hash function* (Schneider, 1996), is a particular type of hash function. It takes as input a string of arbitrary length, and deterministically produces a bit string of fixed length. It is such that it is virtually impossible to reconstruct a message given its digest, and that the probability of collisions, i.e. of two strings being given the same digest, is negligible.

At the end of the initialization, neither Bob nor Tina can access the content of the PT, unless they collude.

2.2 Retrieval

During translation, Bob has a source phrase s and would like to retrieve from the PT the corresponding entry, if it is present. To do so (Fig. 2):

1. Bob computes the digest d of s using the same cryptographic hash function used by Alice in the initialization phase;
2. Bob checks whether $d \in \{d_i\}_{i=1,\ldots,n}$. If the check is negative then s does not have an entry in the PT, and the process stops. If the check is positive then s has an entry in the PT: let i_s be the corresponding index;

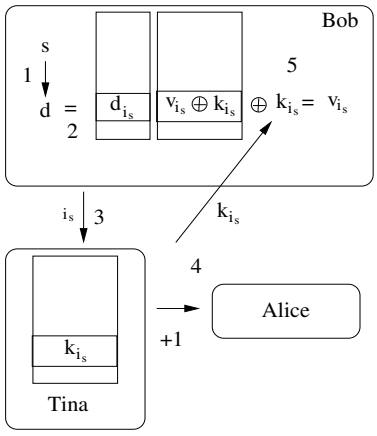

Figure 2: The retrieval phase (Sec. 2.2).

3. Bob requests to Tina key k_{i_s};

4. Tina sends Bob k_{i_s} and notifies Alice, who can increment a counter of PT entries downloaded by Bob;

5. Bob decrypts $v_{i_s} \oplus k_{i_s}$ using key k_{i_s}, and recovers v_{i_s}.

At the end of the process, Bob retrieved from the PT owned by Alice an entry if and only if it matched phrase s (this is guaranteed by the virtual absence of collisions ensured by the cryptographic hash functions used for computing phrase digests). Alice was notified by Tina that Bob downloaded one entry, as desired, while neither Tina nor Alice could learn s, unless they colluded.

3 Implementation

For clarity of exposition, in Section 2.2 we presented a method for looking up PT entries involving one interaction for each phrase look-up. In our implementation, we batch all requests for all source phrases up to a predefined length for all sentences in a given file. This mirrors the standard practice of filtering the phrase table for a given source file to translate before starting the actual decoding.

Out of the large choice of cryptographic hash functions in the literature (Schneider, 1996), we chose 128 bits *md5* for its widespread availability in multiple programming languages and environments.

For encrypting entries, we used bit-wise XOR with a string of random bits (the key) of the same length as the encrypted item. This symmetric encryption is known as *one-time pad*, and it is unbreakable, provided key bits are really random.

Both keys and ciphertext are indexed and sorted by increasing md5 digest of the corresponding source phrase. For retrieving all entries matching a given text file, Bob generates md5 digests for all source phrases up to a maximum length, sorts them, and performs a *join* with the encrypted entry file. Matching digests are then sent to Tina for her to join with the keys. It is important that Bob uses the same tokenizer/word segmentation scheme used by Alice in preprocessing training data before extracting the PT.

Note that it is never necessary to have any massive data structure in main memory, and all process steps except the initial sorting by md5 digest are linear in the number of PT entries or in the number of tokens to look up. The process results however in increased storage and bandwidth requirements, since ciphertext and key have each roughly the same size as the original PT.

4 Related work

We are not aware of any previous work directly addressing the problem we solve, i.e. private access to a phrase table or other resources for the purpose of performing statistical machine translation. Private access to electronic information in general, however, is an active research area. While effective, the scheme proposed here is rather basic, compared to what can be found in specialized literature, e.g. (Chor et al., 1998; Bellovin and Cheswick, 2004). An interesting and relatively recent survey of the field of secure multiparty computation and privacy-preserving data mining is (Lindell and Pinkas, 2009).

5 Experiments

We validated our simple implementation using a phrase table of 38,488,777 lines created with the Moses toolkit[3](Koehn et al., 2007) phrase-based SMT system, corresponding to 15,764,069 entries for distinct source phrases[4].

[3]http://www.statmt.org/moses/

[4]The *birthday bound* for a 128 bit hash like md5 for a collision probability of 10^{-18} is around $2.6 * 10^{10}$. This means

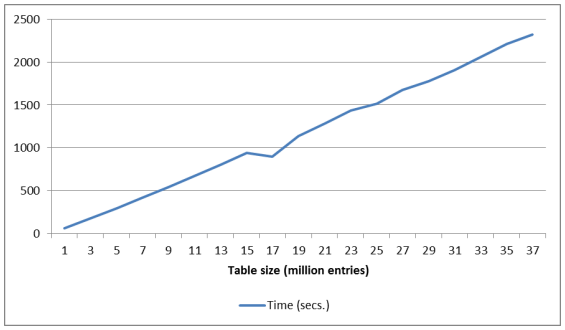

Figure 3: Time required to complete the initialization as a function of the number of lines in the original PT.

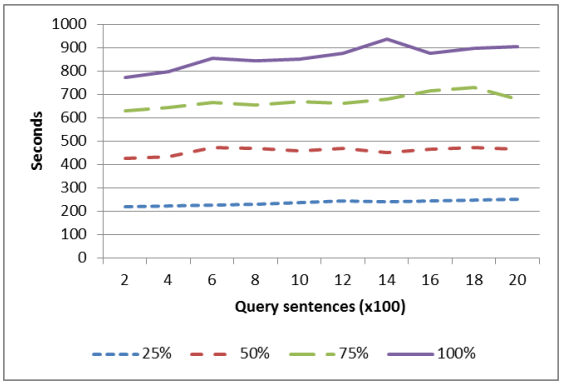

Figure 4: Time required for retrieval as a function of the number of sentences in the query, for different subsets of the original phrase table.

This PT was obtained processing the training data of the English-Spanish Europarl corpus used in the WMT 2008 shared task[5]. We used a 2,000 sentence test set of the same shared evaluation for experimenting with the querying phase.

We conducted all experiments on a single core of an ordinary Linux server[6] with 32Gb of RAM. Both initialization and retrieval can be easily parallelized.

Figure 3 shows the time required to complete the initialization phase as a function of the size of the original PT (in million of lines). The progression is largely linear, and the overall initialization time of roughly 45 minutes for the complete PT indicates that the method can be used in practice. Note that the Europarl corpus originating the phrase-table is much larger than most TMs available at even large language service providers.

Figure 4 displays the time required to complete retrieval for subsets of increasing size of the 2,000 sentence test set, and for phrase tables uniformly sampled at 25%, 50%, 75% and 100%. 217,019 distinct digests are generated for all possible phrase of length up to 6 from the full test set, resulting in the retrieval of 47,072 entries (596,560 lines) from the full phrase table. Our implementation of the retrieval uses the Unix *join* command on the ciphertext and the key tables, and performs a full scan through those files. Complexity hence depends more on the size of the PT than on the length of the query. An ad-hoc indexing of the encrypted entries and of the keys in e.g. a standard database would make the dependency logarithmic in the number of entries, and linear in the number of source tokens. Digests' prefixes are perfectly suited for bucketing ciphertext and keys. This would be useful if query batches are small.

6 Conclusions

Some SMT systems never get deployed because of legitimate and incompatible concerns of the prospective users and of the training data owners. We propose a method that guarantees to the owner of a TM that only some fraction of an artifact derived from the original resource, a phrase-table, is transferred, and only in a very controlled way allowing to track downloads. This same method also guarantees the privacy of the user, who is not required to disclose the content of what needs translation.

Empirical validation on demanding conditions shows that the proposed method is practical on ordinary computing infrastructure.

This same method can be easily extended to other resources used by SMT systems, and indeed even beyond SMT itself, whenever similar constraints on data access exist.

that if the hash distributed keys perfectly uniformly, then about 26 billion entries would be required for the collision probability to exceed 10^{-18}. While no hash function, including md5, distributes keys *perfectly* evenly (Bellare and Kohno, 2004), the number of entries likely to be handled in our application is orders of magnitude smaller than the bound.

[5]http://www.statmt.org/wmt08/shared-task.html

[6]Intel Xeon 3.1 GHz.

References

Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 529–536, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mihir Bellare and Tadayoshi Kohno. 2004. Hash function balance and its impact on birthday attacks. In *Advances in Cryptology, EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 401–418.

Steven M. Bellovin and William R. Cheswick. 2004. Privacy-enhanced searches using encrypted bloom filters. Technical Report CUCS-034-07, Columbia University.

Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. 1998. Private information retrieval. *Journal of the ACM*, 45(6):965–982.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yehuda Lindell and Benny Pinkas. 2009. Secure multiparty computation and privacy-preserving data mining. *The Journal of Privacy and Confidentiality*, 1(1):59–98.

Bruce Schneider. 1996. *Applied Cryptography*. John Wiley and sons.