# Complexity of finding the BLEU-optimal hypothesis in a confusion network

**Gregor Leusch** and **Evgeny Matusov** and **Hermann Ney**
RWTH Aachen University, Germany
{leusch,matusov,ney}@cs.rwth-aachen.de

## Abstract

Confusion networks are a simple representation of multiple speech recognition or translation hypotheses in a machine translation system. A typical operation on a confusion network is to find the path which minimizes or maximizes a certain evaluation metric. In this article, we show that this problem is generally NP-hard for the popular BLEU metric, as well as for smaller variants of BLEU. This also holds for more complex representations like generic word graphs. In addition, we give an efficient polynomial-time algorithm to calculate unigram BLEU on confusion networks, but show that even small generalizations of this data structure render the problem to be NP-hard again.

Since finding the optimal solution is thus not always feasible, we introduce an approximating algorithm based on a multi-stack decoder, which finds a (not necessarily optimal) solution for $n$-gram BLEU in polynomial time.

## 1 Introduction

In machine translation (MT), confusion networks (CNs) are commonly used to represent alternative versions of sentences. Typical applications include translation of different speech recognition hypotheses (Bertoldi et al., 2007) or system combination (Fiscus, 1997; Matusov et al., 2006).

A typical operation on a given CN is to find the path which minimizes or maximizes a certain evaluation metric. This operation can be used in applications like Minimum Error Rate Training (Och, 2003), or optimizing system combination as described by Hillard et al. (2007). Whereas this is easily achievable for simple metrics like the *Word Error Rate* (WER) as described by Mohri and Riley (2002), current research in MT uses more sophisticated measures, like the BLEU score (Papineni et al., 2001). Zens and Ney (2005) first described this task on general word graphs, and sketched a complete algorithm for calculating the maximum BLEU score in a word graph. While they do not give an estimate on the complexity of their algorithm, they note that already a simpler algorithm for calculating the Position independent Error Rate (PER) has an exponential worst-case complexity. The same can be expected for their BLEU algorithm. Dreyer et al (2007) examined a special class of word graphs, namely those that denote constrained reorderings of single sentences. These word graphs have some properties which simplify the calculation; for example, no edge is labeled with the empty word, and all paths have the same length and end in the same node. Even then, their decoder does not optimize the true BLEU score, but an approximate version which uses a language-model-like unmodified precision. We give a very short introduction to CNs and the BLEU score in Section 2.

In Section 3 we show that finding the best BLEU score is an NP-hard problem, even for a simplified variant of BLEU which only scores unigrams and bigrams. The main reason for this problem to become NP-hard is that by looking at bigrams, we allow for one decision to also influence the following decision, which itself can influence the decisions after that. We also show that this also holds for unigram BLEU and the *position independent error rate* (PER) on a slightly augmented variant of CNs which allows for edges to carry multiple symbols. The concatenation of symbols corresponds to the interdependency of decisions in the case of bigram matches above.

NP-hard problems are quite common in machine translation; for example, Knight (1999) has shown that even for a simple form of statistical MT models, the decoding problem is NP-complete. More recently, DeNero and Klein (2008) have proven the NP-completeness of the phrase alignment problem.

But even a simple, common procedure as BLEU scoring, which can be performed in linear time on single sentences, becomes a potentially intractable problem as soon as it has to be performed on a slightly more powerful representation, such as confusion networks. This rather surprising result is the motivation of this paper.

The problem of finding the best unigram BLEU score in an unaugmented variant of CNs is not NP-complete, as we show in Section 4. We present an algorithm that finds such a unigram BLEU-best path in polynomial time.

An important corollary of this work is that calculating the BLEU-best path on general word graphs is also NP-complete, as CNs are a true subclass of word graphs. It is still desirable to calculate a "good" path in terms of the BLEU score in a CN, even if calculating the best path is infeasible. In Section 5, we present an algorithm which can calculate "good" solutions for CNs in polynomial time. This algorithm can easily be extended to handle arbitrary word graphs. We assess the algorithm experimentally on real-world MT data in Section 6, and draw some conclusions from the results in this article in Section 7.

## 2 Confusion networks

A *confusion network* (CN) is a word graph where each edge is labeled with exactly zero or one symbol, and each path from the start node to the end node visits each node of the graph in canonical order. Usually, we represent unlabeled edges by labeling them with the *empty word* $\varepsilon$.

Within this paper, we represent a CN by a list of lists of words $\{w_{i,j}\}$, where each $w_{i,j}$ corresponds to a symbol on an edge between nodes $i$ and $i+1$. A *path* in this CN can be written as a string of integers, $a_1^n = a_1, \ldots, a_n$, such that the path is labeled $w_{1,a_1} w_{2,a_2} \ldots w_{n,a_n}$. Note that there can be a different number of possible words, $j$, for different positions $i$.

### 2.1 BLEU and variants

The BLEU score, as defined by Papineni et al. (2001), is the modified $n$-gram precision of a hypothesis, with $1 \leq n \leq N$, given a set of reference translations $R$. "Modified precision" here means that for each $n$-gram, its maximum number of occurrences within the reference sentences is counted, and only up to that many occurrences in the hypothesis are considered to be correct. The geometric mean over the precisions for all $n$ is calculated, and multiplied by a *brevity penalty* $bp$. This brevity penalty is $1.0$ if the hypothesis sentence is at least as long as the reference sentence (special cases occur if multiple reference sentences with different length exists), and less than $1.0$ otherwise. The exact formulation can be found in the cited paper; for the proofs in our paper it is enough to note that the BLEU score is $1.0$ exactly if all $n$-grams in the hypothesis occur at least that many times in a reference sentence, and if there is a reference sentence which is as long as or shorter than the hypothesis. Assuming that we can always provide a dummy reference sentence shorter than this length, we do not need to regard the brevity penalty in these proofs. Within the following proofs of NP-hardness, we will only require confusion networks (and word graphs) which do not contain empty words, and where all paths from the start node to the end node have the same length.

Usually, in the definition of the BLEU score, $N$ is set to 4; within this article we denote this metric as 4BLEU. We can also restrict the calculations to unigrams only, which would be 1BLEU, or to bigrams and unigrams, which we denote as 2BLEU.

Similar to the 1BLEU metric is the Position independent Error Rate PER (Tillmann et al., 1997), which counts the number of substitutions, insertions, and deletions that have to be performed on the unigram counts to have the hypothesis counts match the reference counts. Unlike 1BLEU, for PER to be optimal (here, $0.0$), the reference counts must match the candidate counts exactly.

Given a CN $\{w_{i,j}\}$ and a set of reference sentences $R$, we define the optimization problem

**Definition 1 (CN-2BLEU-OPTIMIZE)** *Among all paths $a_1^I$ through the CN, what is the path with the highest 2BLEU score?*

Related to this is the decision problem

**Definition 2 (CN-2BLEU-DECIDE)** *Among all paths $a_1^I$ through the CN, is there a path with a 2BLEU score of* $1.0$*?*

Similarly we define CN-4BLEU-DECIDE, CN-PER-DECIDE, etc.

## 3 CN-2BLEU-DECIDE is NP-complete

We now show that CN-2BLEU-DECIDE is NP-complete. It is obvious that the problem is in NP: Given a path $a_1^I$, which is polynomial in size to the problem, we can decide in polynomial time whether $a_1^I$ is a solution to the problem – namely by calculating the BLEU score. We now show that there is a problem known to be NP-complete which can be polynomially reduced to CN-2BLEU-DECIDE. For our proof, we choose 3SAT.

### 3.1 3SAT

Consider the following problem:

**Definition 3 (3SAT)** *Let $X = \{x_1, \dots, x_n\}$ be a set of Boolean variables, let $\mathcal{F} = \bigwedge_{i=1}^{k} (L_{i,1} \vee L_{i,2} \vee L_{i,3})$ be a Boolean formula, where each literal $L_{i,j}$ is either a variable $x$ or its negate $\overline{x}$. Is there a assignment $\beta : X \to \{0, 1\}$ such that $\beta \models \mathcal{F}$? In other words, if we replace each $x$ in $\mathcal{F}$ by $\beta(x)$, and each $\overline{x}$ by $1 - \beta(x)$, does $\mathcal{F}$ become true?*

It has been shown by Karp (1972) that 3SAT is NP-complete. Consequently, if for another problem in NP there is polynomial-size and -time reduction of an arbitrary instance of 3SAT to an instance of this new problem, this new problem is also NP-complete.

### 3.2 Reduction of 3SAT to CN-2BLEU-DECIDE

Let $\mathcal{F}$ be a Boolean formula in 3CNF, and let $k$ be its size, as in Definition 3. We will now reduce it to a corresponding CN-2BLEU-DECIDE problem. This means that we create an alphabet $\Sigma$, a confusion network $\mathcal{C}$, and a set of reference sentences $R$, such that there is a path through $\mathcal{C}$ with a BLEU score of 1.0 exactly if $\mathcal{F}$ is solvable:

Create an alphabet $\Sigma$ based on $\mathcal{F}$ as $\Sigma := \{x_1, \dots, x_n\} \cup \{\overline{x_1}, \dots, \overline{x_n}\} \cup \{\diamond\}$. Here, the $x_i$ and $\overline{x_i}$ symbols will correspond to the variable with the same name or their negate, respectively, whereas $\diamond$ will serve as an "isolator symbol", to avoid unwanted bigram matches or mismatches between separate parts of the constructed CN or sentences.

Consider the CN $\mathcal{C}$ from Figure 1.

Consider the following set of reference sentences:

$$\begin{aligned} R := \{ & \diamond (x_1 \diamond)^k (x_2 \diamond)^k \dots (x_n \diamond)^k \\ & \diamond (\overline{x_1} \diamond)^k (\overline{x_2} \diamond)^k \dots (\overline{x_n} \diamond)^k, \\ & (x_1)^k \diamond (\overline{x_1})^k \diamond \dots (x_n)^k \diamond (\overline{x_n})^k \diamond (\diamond)^{k+n} \ \} \end{aligned}$$

where $(x)^k$ denotes $k$ subsequent occurrences of $x$. Clearly, both $\mathcal{C}$ and $R$ are of polynomial size in $n$ and $k$, and can be constructed in polynomial time.

Then,

*There is an assignment $\beta$ such that $\beta \models \mathcal{F}$*

$\Leftrightarrow$

*There is a path $a_1^I$ through $\mathcal{C}$ such that* $\text{BLEU}(a_1^I, R) = 1.0$.

**Proof: "$\Rightarrow$"**

Let $\beta$ be an assignment under which $\mathcal{F}$ becomes true. Create a path $a_1^I$ as follows: Within $\mathcal{A}$, for each set of edges $L_{i,1}$, $L_{i,2}$, $L_{i,3}$, choose the path through an $x$ where $\beta(x) = 1$, or through an $\overline{x}$ where $\beta(x) = 0$. Note that there must be such an $x$, because otherwise the clause $L_{i,1} \vee L_{i,2} \vee L_{i,3}$ would not be true under $\beta$. Within $\mathcal{B}$, select the path always through $x_i$ if $\beta(x_i) = 0$, and through $\overline{x_i}$ if $\beta(x_i) = 1$.

Then, $a_1^I$ consists of, for each $i$,

- At most $k$ occurrences of both $x_i$ and $\overline{x_i}$
- At most $k$ occurrences of each of the bigrams $x_i\diamond$, $\diamond x_i$, $\overline{x_i}\diamond$, $\diamond\overline{x_i}$, $x_i x_i$, and $\overline{x_i}\,\overline{x_i}$
- No other bigram than those listed above.

For all of these unigram and bigram counts, there is a reference sentence in $R$ which contains at least as many of those unigrams/bigrams as the path. Thus, the unigram and bigram precision of $a_1^I$ is 1.0. In addition, there is always a reference sentence whose length is shorter than that of $a_1^I$, such that the brevity penalty is also 1.0. As a result, $\text{BLEU}(a_1^I, R) = 1.0$.

"$\Leftarrow$"

Let $a_1^I$ be a path through $\mathcal{C}$ such that $BLEU(a_1^I, R) = 1.0$. Because there is no bigram $x_i\overline{x_i}$ or $\overline{x_i}x_i$ in $R$, we can assume that for each $x_i$, either only $x_i$ edges, or only $\overline{x_i}$ edges appear in the $\mathcal{B}$ part of $a_1^I$, each at most $k$ times. As no unigram $x_i$ and $\overline{x_i}$ appears more than $k$ times in $R$, we can assume that, if the $\overline{x_i}$ edges are passed in $\mathcal{B}$, then only the $x_i$ edges are passed in $\mathcal{A}$, and vice versa. Now, create an assignment $\beta$ as follows:

$$\beta := \begin{cases} 0 & \text{if } x_i \text{ edges are passed in } \mathcal{B} \\ 1 & \text{otherwise} \end{cases}$$

Then, $\beta \models \mathcal{F}$. Proof: Assume that $\mathcal{F}_\beta = 0$. Then there must be a clause $i$ such that $L_{i,1} \vee L_{i,2} \vee L_{i,3} =$

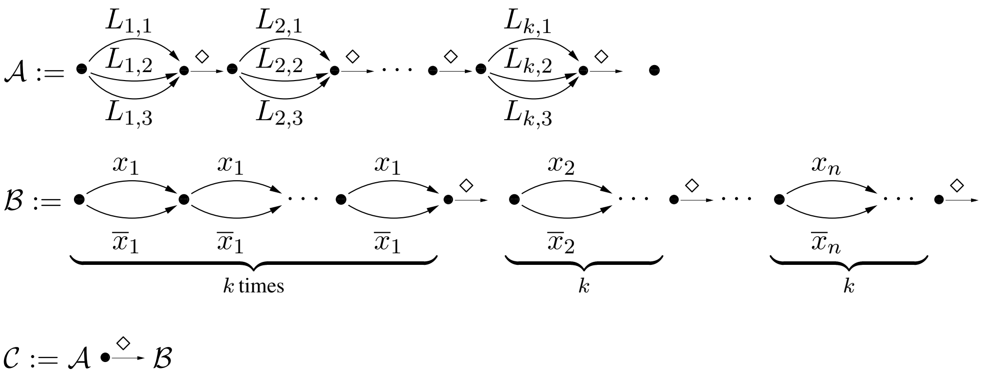


Figure 1: CN constructed from a 3SAT formula $\mathcal{F}$. $\mathcal{C}$ is the concatenation of the left part $\mathcal{A}$, and the right path $\mathcal{B}$, separated by an isolating $\diamond$.

0. At least one of the edges $L_{i,j}$ associated with the literals of this clause must have been passed by $a_1^K$ in $\mathcal{A}$. This literal, though, can not have been passed in $\mathcal{B}$. As a consequence, $\beta(L_{i,j}) = 1$. But this means that $L_{i,1} \vee L_{i,2} \vee L_{i,3} = 1 \rightsquigarrow$ contradiction.

Because CN-2BLEU-DECIDE is in NP, and we can reduce an NP-complete problem (3SAT) in polynomial time to a CN-2BLEU-DECIDE problem, this means that CN-2BLEU-DECIDE is NP-complete.

### 3.3 CN-4BLEU-DECIDE

It is straightforward to modify the construction above to create an equivalent CN-4BLEU-DECIDE problem instead: Replace each occurrence of the isolating symbol $\diamond$ in $\mathcal{A}, \mathcal{B}, \mathcal{C}, R$ by three consecutive isolating symbols $\diamond\diamond\diamond$. Then, everything said about unigrams still holds, and bi-, tri- and fourgrams are handled equivalently: Previous unigram matches on $\diamond$ correspond to uni-, bi-, and trigram matches on $\diamond$, $\diamond\diamond$, $\diamond\diamond\diamond$. Bigram matches on $x\diamond$ correspond to bi-, tri-, and fourgram matches on $x\diamond$, $x\diamond\diamond$, $x\diamond\diamond\diamond$, and similar holds for bigram matches $\overline{x}\diamond$, $\diamond x$, $\diamond\overline{x}$. Unigram matches $x$, $\overline{x}$, and bigram matches $xx$ etc. stay the same. Consequently, CN-4BLEU-DECIDE is also an NP-complete problem.

### 3.4 CN*-1BLEU-DECIDE

Is it possible to get rid of the necessity for bigram counts in this proof? One possibility might be to look at slightly more powerful graph structures, CN*. In these graphs, each edge can be labeled by arbitrarily many symbols (instead of just zero or one). Then, consider a CN* graph $\mathcal{C}' := \mathcal{A}\bullet\xrightarrow{\diamond}\mathcal{B}'$, with $\mathcal{B}'$ as in Figure 2.

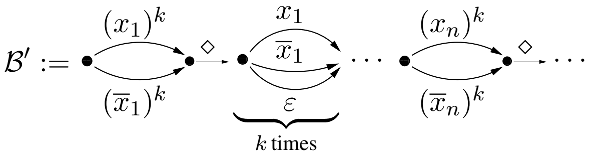


Figure 2: Right part of a CN* constructed from a 3SAT formula $\mathcal{F}$.

With

$$R' := \{(x_1)^k(\overline{x_1})^k \ldots (x_n)^k(\overline{x_n})^k(\diamond)^k\}$$

we can again assume that either $x_i$ or $\overline{x_i}$ appears $k$ times in the $\mathcal{B}'$-part of a path $a_1^K$ with $1BLEU(a_1^K, R') = 1.0$, and that for every solution $\beta$ to $\mathcal{F}$ there is a corresponding path $a_1^K$ through $\mathcal{C}'$ and vice versa. In this construction, we also have exact matches of the counts, so we can also use PER in the decision problem.

While CN* are generally not word graphs by themselves due to the multiple symbols on edges, it is straightforward to create an equivalent word graph from a given CN*, as demonstrated in Figure 3. Consequently, deciding unigram BLEU and unigram PER are NP-complete problems for general word graphs as well.

## 4 Solving CN-1BLEU-DECIDE in polynomial time

It is not a coincidence that we had to resort to bigrams or to edges with multiple symbols for NP-completeness: It turns out that CN-1BLEU-DECIDE, where the order of the words does not

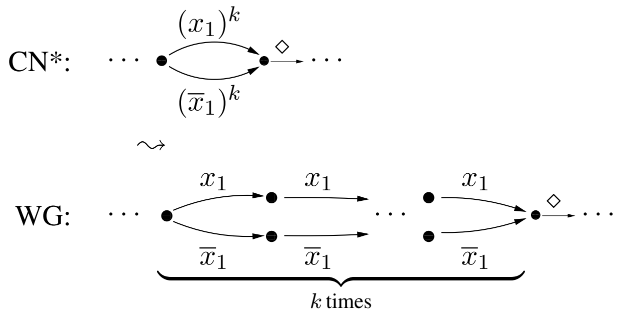
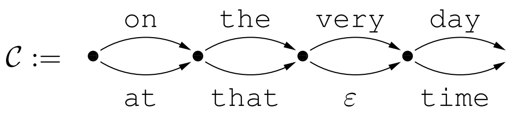


Figure 3: Construction of a word graph from a CN* as in $\mathcal{B}'$.

matter at all, can be decided in polynomial time using the following algorithm, which disregards a brevity penalty for the sake of simplicity:

Given a vocabulary $X$, a CN $\{w_{i,j}\}$, and a set of reference sentences $R$ together with their unigram BLEU counts $c(x) : X \to \mathbb{N}$ and $C := \sum_{x \in X} c(x)$,

1. Remove all parts from $w$ where there is an edge labeled with the empty word $\varepsilon$. This step will always increase unigram precision, and can not hurt any higher $n$-gram precision here, because $n = 1$. In the example in Figure 4, the edges labeled `very` and $\varepsilon$ respectively are affected in this step.
2. Create nodes $A_0 := \{1, \ldots, n\}$, one for each node with edges in the CN. In the example in Figure 5, the three leftmost column heads correspond to these nodes.
3. Create nodes $B := \{x.j \mid x \in X, 1 \leq j \leq c(x)\}$. In other words, create a unique node for each "running" word in $R$ – e.g. if the first and second reference sentence contain $x$ once each, and the third reference contains $x$ twice, create exactly $x.1$ and $x.2$. In Figure 5, those are the row heads to the right.
4. Fill $A$ with empty nodes to match the total length: $A := A_0 \cup \{\varepsilon.j \mid 1 \leq j \leq C - n\}$. If $n > C$, the BLEU precision can not be 1.0. The five rightmost columns in Figure 5 correspond to those.
5. Create edges $E := \{(i, w_{i,j}.k) \mid 1 \leq i \leq n, \text{ all } j, 1 \leq c(w_{i,j})\} \cup \{(i, \varepsilon.j) \mid 1 \leq i \leq n, \text{ all } j\}$. These edges are denoted as ○ or ● in Figure 5.



$R := \{$ `on the same day,`
`at the time and the day` $\}$

Figure 4: Example for CN-1BLEU-DECIDE.

| 1 | 2 | 3 | $\varepsilon.1$ | $\varepsilon.2$ | $\varepsilon.3$ | $\varepsilon.4$ | $\varepsilon.5$ | |
|---|---|---|---|---|---|---|---|---|
| ● | | | ○ | ○ | ○ | ○ | ○ | `on` |
| | ● | | ○ | ○ | ○ | ○ | ○ | `the.1` |
| | ○ | | ● | ○ | ○ | ○ | ○ | `the.2` |
| | | | ○ | ● | ○ | ○ | ○ | `same` |
| | | ● | ○ | ○ | ○ | ○ | ○ | `day` |
| ○ | | | ○ | ○ | ● | ○ | ○ | `at` |
| | | ○ | ○ | ○ | ○ | ● | ○ | `time` |
| | | | ○ | ○ | ○ | ○ | ● | `and` |

Figure 5: Bipartite graph constructed to find the optimal 1BLEU path in Figure 4. One possible maximum bipartite matching is marked with ●.

6. Find the *maximum bipartite matching* $M$ between $A$ and $B$ given $E$. Figure 5 shows such a matching with ●.
7. If all nodes in $A$ and $B$ are covered by $M$, then 1BLEU$(\{w_{i,j}\}, R) = 1.0$. The words that are matched to $A_0$ then form the solution path through $\{w_{i,j}\}$.

Figure 4 gives an example of a CN and a set of references $R$, for which the best 1BLEU path can be constructed by the algorithm above. The bipartite graph constructed in Step 1 to Step 4 for this example, given in matrix form, can be found in Figure 5.

Such a solution to Step 6, if found, corresponds exactly to a path through the confusion network with 1BLEU=1.0, and vice versa: for each position $1 \leq i \leq n$, the matched word corresponds to the word that is selected for the position of the path; "surplus" counts are matched with $\varepsilon$s.

Step 6 can be performed in polynomial time (Hopcroft and Karp, 1973) $O((C+n)^{5/2})$; all other steps in linear time $O(C+n)$. Consequently, CN-1BLEU can be decided in polynomial time $O((C+n)^{5/2})$. Similarly, an actual optimum 1BLEU score can be calculated in $O((C+n)^{5/2})$.

It should be noted that the only alterations in the hypothesis length, and as a result the only alterations in the brevity penalty, will come from Step 1. Consequently, the brevity penalty can be taken into account as follows: Consider that there are $M$ nodes with an empty edge in $\{w_{i,j}\}$. Instead of removing them in Step 1, keep them in, but for each $1 \leq m \leq M$, run through steps 2 to 6, but add $m$ nodes $\varepsilon.1, \ldots, \varepsilon.m$ to $B$ in Step 3, and add corresponding edges to these nodes to $E$ in Step 5. After each iteration (which leads to a constant hypothesis length), calculate precision and brevity penalty. Select the best product of precision and brevity penalty in the end. The overall time complexity now is in $M \cdot O((C+n)^{5/2})$.

A PER score can be calculated in a similar fashion.

## 5 Finding approximating solutions for CN-4BLEU in polynomial time

Knowing that the problem of finding the BLEU-best path is an NP-complete problem is an unsatisfactory answer in practice – in many cases, having a good, but not necessarily optimum path is preferable to having no good path at all.

A simple approach would be to walk the CN from the start node to the end node, keeping track of $n$-grams visited so far, and choosing the word next which maximizes the $n$-gram precision up to this word. Track is kept by keeping $n$-gram count vectors for the hypothesis path and the reference sentences, and update those in each step.

The main problem with this approach is that often the local optimum is suboptimal on the global scale, for example if a word occurs on a later position again.

Zens and Ney (2005) on the other hand propose to keep all $n$-gram count vectors instead, and only recombine path hypotheses with identical count vectors. As they suspect, the search space can become exponentially large.

In this paper, we suggest a compromise between these two extremes, namely keeping active a sufficiently large number of "path hypotheses" in terms of $n$-gram precision, instead of only the first best, or of all. But even then, edges with empty words pose a problem, as stepping along an empty edge will never decrease the precision of the local path. In certain cases, steps along empty edges may affect the $n$-gram precision for higher $n$-grams. But this will only take effect after the next non-empty step, it does not influence the local decision in a node. Stepping along a non-empty edge will often decrease the local precision, though. As a consequence, a simple algorithm will prefer paths with shorter hypotheses, which leads to a suboptimal total BLEU score, because of the brevity penalty. One can counter this problem for example by using a brevity penalty already during the search. But this is problematic as well, because it is difficult to define a proper partial reference length in this case.

The approach we propose is to compare only partial path hypotheses with the same number of empty edges, and ending in the same position in the confusion network. This idea is illustrated in Figure 6: We compare only the partial precision of path hypotheses ending in the same node. Due to the simple nature of this search graph, it can easily be traversed in a left-to-right, top-to-bottom manner. With regard to a node currently being expanded, only the next node in the same row, and the corresponding columns in the next row need to be kept active. When implementing this algorithm, Hypotheses should be compared on the modified BLEUS precision by Lin and Och (2004) because the original BLEU precision equals zero as long as there are no higher $n$-gram matches in the partial hypotheses, which renders meaningful comparison hard or impossible.

In the rightmost column, all path hypotheses within a node have the same hypothesis length. Consequently, we can select the hypothesis with the best (brevity-penalized) BLEU score by multiplying the appropriate brevity penalty to the precision of the best path ending in each of these nodes. If we always expand all possible path hypotheses within the nodes, and basically run a full search, we will always find the BLEU-best path this way. From the proof above, it follows that the number of path hypothesis we would have to keep can become exponentially large. Fortunately, if a "good" solution is good enough, we do not have to keep all possible path hypotheses, but only the $S$ best ones for a given constant $S$, or those with a precision not worse than $c$ times the precision of the best hypothesis within the node. Assuming that adding and removing an element to/from a size-limited stack of size $S$ takes time $O(\log S)$, that we allow at most $E$ empty edges in a solution, and that there are $j$ edges in each of the $n$ positions, this algorithm has a time complexity of

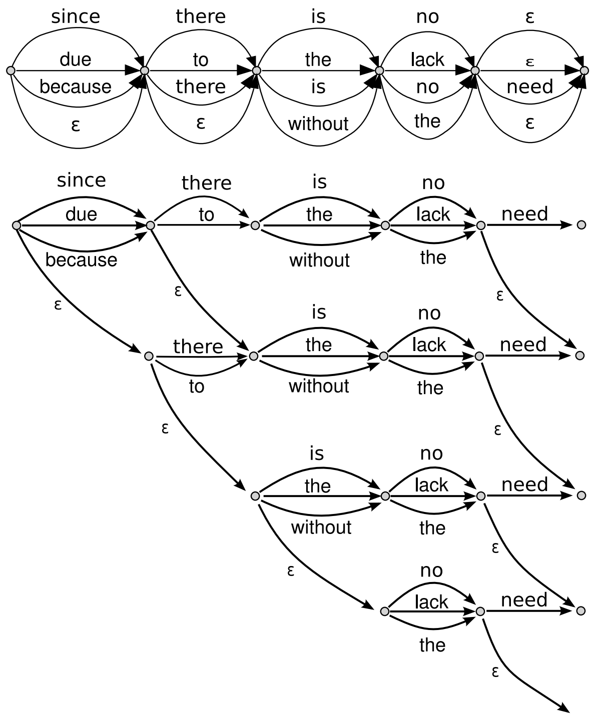


Figure 6: Principle of the multi-stack decoder used to find a path with a good BLEU score. The first row shows the original confusion network, the following rows show the search graph. Duplicate edges were removed, but no word was considered "unknown".

$O(E \cdot n \cdot j \cdot S \log S)$.

To reduce redundant duplicated path hypotheses, and by this to speed up the algorithm and reduce the risk that good path hypotheses are pruned, the confusion network should be *simplified* before the search, as shown in Figure 6:

1. Remove all words in the CN which do not appear in any reference sentence, if there at least one "known" non-empty word at the same position. If there is no such "known" word, replace them all by a single token denoting the "unknown word".

2. Remove all duplicate edges in a position, that is, if there are two or more edges carrying the same label in one position, remove all but one for them.

These two steps will keep at least one of the BLEU-best paths intact. But they can remove the average branching factor ($j$) of the CN significantly, which leads to a significantly lower number of duplicate path hypotheses during the search.

Table 1: Statistics of the (Chinese–)English MT corpora used for the experiments

| | NIST 2003 | NIST 2006 | |
|---|---|---|---|
| number of systems | 4 | 4 | |
| number of ref. | 4 | 4 | per sent. |
| sentences | 919 | 249 | |
| system length | 28.4 | 33.2 | words* |
| ref. length | 27.5 | 34.2 | words* |
| best path | 24.4 | 33.9 | words* |
| CN length | 40.7 | 39.5 | nodes* |
| best single system | 29.3<br>30.5 | 52.5<br>51.6 | BLEU<br>BLEUS* |

*average per sentence

Our algorithm can easily be extended to handle arbitrary word graphs instead of confusion networks. In this case, each "row" in Figure 6 will reflect the structure of the word graph instead of the "linear" structure of the CN.

While this algorithm searches for the best path for a single sentence only, a common task is to find the best BLEU score over a whole test set – which can mean suboptimal BLEU scores for individual sentences. This adds an additional combinatorial problem over the sentences to the actual decoding process. Both Zens and Ney (2005) and Dreyer et al (2007) use a greedy approach here; the latter estimated the impact of this to be insignificant in random sampling experiments. In our experiments, we used the per-sentence BLEUS score as (greedy) decision criterion, as this is also the pruning criterion. One possibility to adapt this approach to Zens's/Dreyer's greedy approach for system-level BLEU scores might be to initialize $n$-gram counts and hypothesis length not to zero at the beginning of each sentence, but to those of the corpus so far. But as this diverts from our goal to optimize the sentence-level scores, we have not implemented it so far.

## 6 Experimental assessment of the algorithm

The question arises how many path hypotheses we need to retain in each step to obtain optimal paths. To examine this, we created confusion networks out of the translations of the four best MT systems of

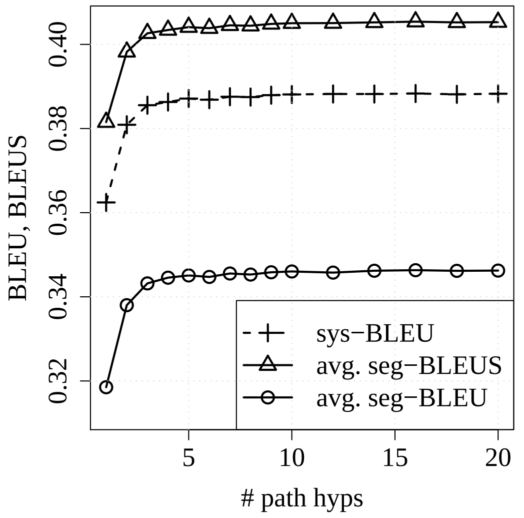


Figure 7: Average of the sentence-wise BLEU and BLEUS score and the system-wide BLEU score versus the number of path hypotheses kept per node during the search. NIST MT03 corpus.

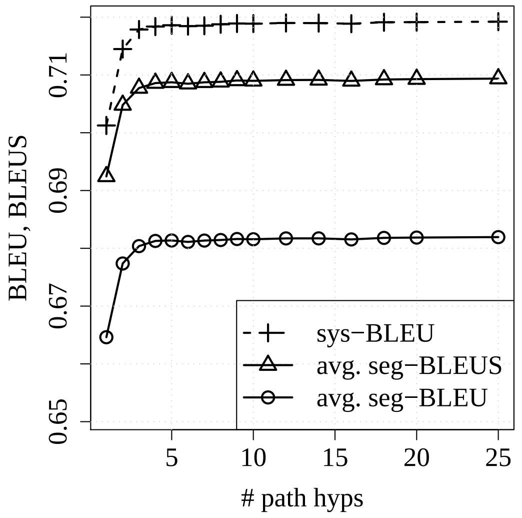


Figure 8: Average of the sentence-wise BLEU and BLEUS score and the system-wide BLEU score versus the number of path hypotheses kept per node during the search. NIST MT06 corpus.

the NIST 2003 and 2006 Chinese–English evaluation campaigns, as available from the Linguistic Data Consortium (LDC). The hypotheses of the best single system served as skeleton, those of the three remaining systems were reordered and aligned to the skeleton hypothesis. This corpus is described in Table 1. Figures 7 and 8 show the measured BLEU scores in three different definitions, versus the maximum number of path hypotheses that are kept in each node of the search graph. Shown are the average sentence-wise BLEUS score, which is what the algorithm actually optimizes, for comparison the average sentence-wise BLEU score, and the total document-wise BLEU score.

All scores increase with increasing number of retained hypotheses, but stabilize around a total of 15 hypotheses per node. The difference over a greedy approach, which corresponds to a maximum of one hypothesis per node if we leave out the separation by path length, is quite significant. No further improvements can be expected for a higher number of hypotheses, as experiments up to 100 hypotheses show.

## 7 Conclusions

In this paper, we showed that deciding whether a given CN contains a path with a BLEU score of 1.0 is an NP-complete problem for $n$-gram lengths $\geq 2$. The problem is also NP-complete if we only look at unigram BLEU, but allow for CNs where edges may contain multiple symbols, or for arbitrary word graphs. As a corollary, any proposed algorithm to find the path with an optimal BLEU score in a CN, even more in an arbitrary word graph, which runs in worst case polynomial time can only deliver an approximation[1].

We gave an efficient polynomial time algorithm for the simplest variant, namely deciding on a unigram BLEU score for a CN. This algorithm can easily be modified to decide on the PER score as well, or to calculate an actual unigram BLEU score for the hypothesis CN.

Comparing these results, we conclude that the ability to take bi- or higher $n$-grams into account, be it in the scoring (as in 2BLEU), or in the graph structure (as in CN*), is the key to render the problem NP-hard. Doing so creates long-range dependencies, which oppose local decisions.

We also gave an efficient approximating algorithm for higher-order BLEU scores. This algorithm is based on a multi-stack decoder, taking into account the empty arcs within a path. Experimental results on real-world data show that our method is indeed able to find paths with a significantly better

[1] provided that $P \neq NP$, of course.

BLEU score than that of a greedy search. The resulting BLEUS score stabilizes already on a quite restricted search space, showing that despite the proven NP-hardness of the exact problem, our algorithm can give useful approximations in reasonable time. It is yet an open problem in how far the problems of finding the best paths regarding a sentence-level BLEU score, and regarding a system-level BLEU score correlate. Our experiments here suggest a good correspondence.

## 8 Acknowledgments

This paper is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023. The proofs and algorithms in this paper emerged while the first author was visiting researcher at the Interactive Language Technologies Group of the National Research Council (NRC) of Canada, Gatineau. The author wishes to thank NRC and Aachen University for the opportunity to jointly work on this project.

## References

Nicola Bertoldi, Richard Zens, and Marcello Federico. 2007. Speech translation by confusion network decoding. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1297–1300, Honululu, HI, USA, April.

John DeNero and Dan Klein. 2008. The complexity of phrase alignment problems. In *Human Language Technologies 2008: The Conference of the Association for Computational Linguistics, Short Papers*, pages 25–28, Columbus, Ohio, June. Association for Computational Linguistics.

Markus Dreyer, Keith Hall, and Sanjeev Khudanpur. 2007. Comparing Reordering Constraints for SMT Using Efficient BLEU Oracle Computation. In *AMTA Workshop on Syntax and Structure in Statistical Translation (SSST) at the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pages 103–110, Rochester, NY, USA, April.

Jonathan G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (ROVER). In *Proceedings 1997 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 347–352, Santa Barbara, CA.

Dustin Hillard, Björn Hoffmeister, Mari Ostendorf, Ralf Schlüter, and Hermann Ney. 2007. iROVER: Improving system combination with classification. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 65–68, Rochester, New York, April.

John E. Hopcroft and Richard M. Karp. 1973. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231.

Richard M. Karp. 1972. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615, December.

Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluation automatic evaluation metrics for machine translation. In *Proc. COLING 2004*, pages 501–507, Geneva, Switzerland, August.

Evgeny Matusov, Nicola Ueffing, and Hermann Ney. 2006. Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. In *Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–40, Trento, Italy, April.

Mehryar Mohri and Michael Riley. 2002. An efficient algorithm for the n-best-strings problem. In *Proc. of the 7th Int. Conf. on Spoken Language Processing (ICSLP'02)*, pages 1313–1316, Denver, CO, September.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

Kishore A. Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, September.

Christoph Tillmann, Stephan Vogel, Hermann Ney, Alex Zubiaga, and Hassan Sawaf. 1997. Accelerated DP based search for statistical translation. In *European Conf. on Speech Communication and Technology*, pages 2667–2670, Rhodes, Greece, September.

Richard Zens and Hermann Ney. 2005. Word graphs for statistical machine translation. In *43rd Annual Meeting of the Assoc. for Computational Linguistics: Proc. Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, pages 191–198, Ann Arbor, MI, June.