Evaluating a Dialog Language Generation System: Comparing the MOUNTAIN System to Other NLG Approaches

Brian Langner, Stephan Vogel, Alan W Black

Language Technologies Institute
Carnegie Mellon University, Pittsburgh, USA
{blangner,vogel,awb}@cs.cmu.edu

Abstract

This paper describes the MOUNTAIN language generation system, a fully-automatic, data-driven approach to natural language generation aimed at spoken dialog applications. MOUNTAIN uses statistical machine translation techniques and natural corpora to generate human-like language from a structured internal language, such as a representation of the dialog state. We briefly describe the training process for the MOUNTAIN approach, and show results of automatic evaluation in a standard language generation domain: the METEO weather forecasting corpus. Further, we compare output from the MOUNTAIN system to several other NLG systems in the same domain, using both automatic and human-based evaluation metrics; our results show our approach is comparable in quality to other advanced approaches. Finally, we discuss potential extensions, improvements, and other planned tests.

Index Terms: natural language generation, evaluation, translation-based generation, weather forecasts, spoken dialog

1. Introduction

In recent years, there has been increasing interest in using data-driven approaches in spoken dialog research. Nearly all of the typical components of a dialog system have had some effort made to use machine learning to improve them; these are nicely summarized in [1]. It seems, though, that trainable language generation for dialog has seen comparatively less work than other modules like ASR, dialog management, and related areas like user simulation. Given the general success of these efforts in related areas, we feel it is likely that such an approach can also work well for language generation.

Because language generation for dialog systems has several key differences from general text generation [2], we feel a dialog system should have language generation that is tailored for its needs. In particular, for many dialog platforms, generation is primarily only for realizing natural language surface forms that correspond to some internal state; often this is referred to as *tactical* generation. In practical terms, what is done is to convert the machine's representation of the dialog state into fluent and natural-sounding sentences. If one thinks of the dialog state representations as a highly structured (and possibly simplistic) language, then this task can be viewed as a *translation* problem, where the goal is to translate from the highly structured internal language of states to fluent and natural English (or any other language) sentences. Wong and Mooney describe a language generation system that uses machine translation techniques [3]; the approach effectively is the inverse of a semantic parser.

Corpus- and statistically-based approaches to language generation, however, have been around for some time now, though only recently have they been starting to be applied to dialog NLG. The Nitrogen generation system [4], for example, derived statistical information from a corpus in order to rank and select grammar-generated surface forms. Ratnaparkhi [5] describes a generation system that can be trained from an annotated corpus, able to produce surface forms using only a semantic representation. Marciniak and Strube [6] describe using a corpus annotated with semantic and grammatical information, which is then used as a linguistic knowledge base for generation. A similar approach as what we propose has used statistical translation methods for summarization and headline generation [7], with some degree of success. Many of these approaches use significant amounts of linguistic knowledge, either from annotations or trained from data, in order to improve the natural language output they produce. The work we describe here attempts to use a broadly similar method – automatically learning generation output from a corpus of examples – but without any explicit linguistic annotation of the corpus.

2. The MOUNTAIN Generation System

We call our approach the MOUNTAIN language generation system [8]: a **m**achine **t**ranslation approach for **n**atural **l**anguage **g**eneration. Our implementation relies on the Moses machine translation system[1], because it and its support tools are freely available, though nothing in our approach is restricted to this specific engine. MOUNTAIN is designed as a fully-automatic, data-driven approach to language generation, targetting applications such as spoken dialog systems. It requires only a parallel corpus of states as an internal language that are aligned with corresponding natural language surface forms. Obtaining a corpus can be done in several different ways, from eliciting examples from a group of people in a separate collection task, to having system developers themselves create the corpus in a similar process as template-writing. With the parallel corpus, the Moses tools are used to train a translation model which is capable of translating from the structured internal language to appropriate surface forms. Additionally, the natural language corpus is used to train a language model for the target language.

As described above, the internal language MOUNTAIN uses can be a structured representation of the dialog state – what the dialog manager intends to convey to the user. For many applications, one can imagine several possible representations for sentences in the internal language. Since we control that, it is important to choose a representation that is suitable for the translation engine; for example, a representation that has severe length mismatches between its internal and target sentences would be a poor choice, since the translation engine does not handle such a condition particularly well.

Once the model is trained, it can be tuned using a separate, held out development set. Using minimum error rate tuning,

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

```
Raw Input:   [[1,_W-SW,10,15,-,-,0600],[2,-,18,22,-,-,1500],[3,_W,20,25,-,-,0000]]
Tokenized:   t1 W-SW 10 15 - - 0600 t2 - 18 22 - - 1500 t3 W 20 25 - - 0000
Output:      W-SW 10-15 INCREASING 18-22 BY MID AFTERNOON THEN VEERING W 20-25 BY LATE EVENING
```

Figure 1: Example of the raw vector of 7-tuples and the corresponding tokenized form used as input text by MOUNTAIN.

more optimal weights can be learned for the language and translation models. Other parameters can be modified or tuned to improve performance as well, such as the distortion limit or length penalty. After models have been trained and tuned, MOUNTAIN uses the translation engine to generate output utterances, given "sentences" from the internal language. Moses uses the trained models to translate into the target natural language; the resulting output is the best result from the translation engine. Note that the output can be novel sentences, not just examples taken from the training data.

It should be noted that the entire process used by MOUNTAIN, from training to generation, does not require any specific linguistic analysis or domain knowledge, and thus can be considered a domain-independent approach. In fact, MOUNTAIN is also *language*-independent, provided the target language is able to be used by the training tools (such as the tokenizer and language model trainer). However, domain knowledge can be useful in some areas when it is available, such as in defining the structure of the internal language.

3. Testing with a Common Domain

3.1. Weather Forecasting: The SUMTIME-METEO Corpus

While an informal examination of MOUNTAIN output seems to show it is a useful approach to language generation [8], capable of generating acceptable responses, it is clear that a more structured evaluation is required. It would also be useful to compare MOUNTAIN's performance to other NLG systems – preferably in an easily-comparable domain. The SUMTIME-METEO corpus [9] has been used by multiple different generation systems [10, 11], and offers an ideal opportunity to compare against other approaches in NLG. This corpus consists of wind and precipitation forecasts written by three different human weather forecasters, along with the weather data used by the human forecasters to create them. Specifically, we wanted to compare our system to those used by Belz and Kow [12], which compared 10 different approaches using the wind forecast data in METEO. This subset of the corpus extracts the wind forecast statements as well as the wind data in the form of vectors of 7-tuple numeric data, ultimately forming an aligned parallel corpus suitable for corpus-based NLG. Overall, the corpus contains 465 entries with data and a corresponding forecast text.

3.2. Training

We used the training process as described above; the METEO parallel corpus was tokenized and case-normalized, and then used to train a phrase model and an English language model. Each 7-tuple in the input data vectors has a sequential ID for the forecast period, wind direction, minimum speed, maximum speed, minimum gust speed, maximum gust speed, and timestamp. If a value is missing or not relevant for a particular 7-tuple, it is represented with a single dash. Each 7-tuple has its values listed in order and separated by commas, with the entire block enclosed in square brackets. Each forecast can have multiple 7-tuples associated with it, with individual 7-tuples separated by commas. We considered using the raw vectors in MOUNTAIN with only the default tokenizer, but found that a custom tokenization would be better suited for training a translation model. Our tokenization strips all square brackets and underscores, replaces commas with spaces, and adds a disambiguating character to the 7-tuple ID token. Figure 1 shows an example of both the raw and the tokenized input.

3.3. Evaluation

The previous evaluations using this corpus report NIST and BLEU scores for automatic measures; both metrics effectively measure n-gram agreement between test and reference strings. Thus, we report those in our evaluation, though we note that NIST is heavily weighted to unigram recall and may not be an ideal measure of adequacy, fluency, or human preference for NLG. We also are reporting METEOR [13] scores, as a different machine translation metric whose design may be more suitable for NLG evaluation.

The METEO corpus has pre-defined splits of 5 overlapping folds for cross-validation, which we have used. This necessitates training (and then testing) what amounts to 5 separate systems; the average of scores from all 5 systems will give a reasonably unbiased view of system performance. Like previous evaluations with this corpus, our reported scores are the 5-fold averages. We tested MOUNTAIN using the test sets distributed in the METEO corpus. Our system used the default Moses models from the training process, without any tuning. We also tried several different things in an effort to improve the baseline results, including minimum error rate tuning, changing the distortion limit and length penalty in the translation model, and training a larger n-gram language model. Most of these had no impact on the resulting scores in this domain, except for a small negative effect from minimum error rate tuning. We believe this is due to the limited amount of training data available to use for tuning in the METEO corpus; the sparsity causes overfitting to the tuning data.

3.4. Comparing to Other Systems

Not much can be said about performance in isolation, however, so a comparison to other systems would be helpful. The corpus distribution also includes the outputs from the 10 previously-

System	NIST	BLEU	METEOR
corpus	9.307	1.000	1.000
PSCFG-sem	7.072	.6352	.8159
PSCFG-unstruc	6.881	.6259	.8081
PCFG-greedy	6.632	.5968	.7969
MOUNTAIN	6.517	.5773	.7765
SUMTIME-hyb	6.086	.5252	.6889
PCFG-2gram	5.468	.4862	.6867
PCFG-viterbi	5.462	.4864	.6863
PCFG-roulette	5.906	.4617	.6853
PBSMT-unstruc	5.684	.4863	.4877
PBSMT-struct	4.186	.3143	.4173
PCFG-random	3.368	.2069	.2102

Table 1: Results of automatic measures for MOUNTAIN compared to the NLG systems and original human-written forecasts used by Belz and Kow [12].

evaluated systems [12], making direct comparison of scores a straightforward task. Table 1 shows the results of multiple different generation systems in the METEO domain. Overall, MOUNTAIN performs fairly well according to automatic measures compared to other systems, surpassing other SMT-based approaches, many of the weaker CFG-based systems, as well as the hand-crafted human-designed SUMTIME-hybrid system. The only systems which are clearly better than MOUNTAIN are the PSCFG-based approaches [3]; these approaches are not fully automatically trained like MOUNTAIN is, however. All three of the automatic metrics are fairly consistent in how systems are ranked relative to each other.

4. Do Automatic Measures Correlate with Ratings by Humans?

One of the key issues with NLG evaluation is its expense, primarily due to the high costs of human-based evaluation. Automatic measures are cheap and simple to use, but it isn't clear that they measure the same things. Thus, it would be helpful to determine what correlation, if any, there is for these automatic measures and human judgments. The METEO corpus, with outputs from multiple systems, also provides an additional opportunity to test the performance and correlation of automatic metrics compared to human judgments, continuing earlier work in that regard [14, 15].

For our evaluation, human evaluators were shown forecast texts and asked to rate them on a 7-point Likert scale for both clarity and readability. These terms were explicitly defined: clarity refers to how clear and understandable a forecast is, and readability refers to how fluent and easy to read a forecast is. In addition to the MOUNTAIN approach, we used 5 systems from the earlier evaluation [12]: both the SMT-based systems, the most consistent CFG-based system (PSCFG-semantic), the handcrafted SUMTIME-hybrid system, and the original human-written corpus. The choice of these systems was made because they were either similar to the MOUNTAIN approach, strong performers in the previous evaluation, or a human-generated standard; additionally they encompass widely varying approaches, from manually annotated to fully automatic. They also cover the entire range of systems in the earlier evaluation, from highly- to poorly-performing, so our results should be easy to compare.

We used 12 randomly selected forecast dates (taken from each fold of the corpus), and included outputs from all 6 systems, resulting in an evaluation set with 72 distinct forecast texts. Raters were presented with 2 different forecast texts from each system, presented in random order, and told to rate on clarity and readability as defined above. Because earlier work in this domain demonstrated that non-experts produced ratings that were highly correlated to those given by domain experts [16], we did not attempt to find weather experts to rate the forecasts. A total of 38 raters completed this task.

4.1. Results

Figure 2 shows the mean clarity and readability scores for each system. Though it matches the results in [12], we are still surprised that the original human corpus is rated poorer than most of the machine-generated texts. The handcrafted SUMTIME system has the highest ratings in both categories, though it is by far the most expensive system in terms of creation effort. The MOUNTAIN approach is significantly better than the other SMT-based systems, and in fact is rated slightly higher than the natural corpus. It is not quite as good as the SUMTIME or PSCFG systems; however, it should be noted that those systems include linguistic knowledge and are not fully automatic like MOUNTAIN. If MOUNTAIN can exploit that as well, it is likely to see a performance improvement.

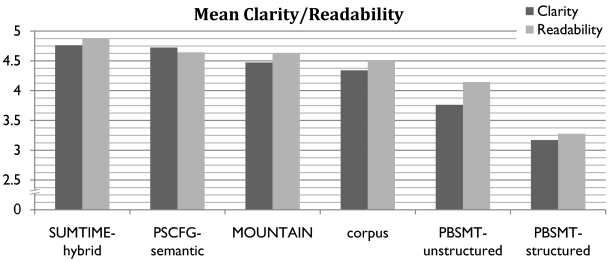

Figure 2: Mean clarity and readability ratings from human evaluation of 5 NLG systems and the original human-written forecasts in the METEO domain.

Table 2 shows both automatic and human-based scores for all the systems based on the 12 forecast texts used in the human evaluation. This is a different test set than reported previously (see Table 1), which explains the different values for the automatic metrics. Ignoring the "corpus" system, which is guaranteed to have a perfect score from the automatic measures, there is definite similarity in rankings between the human and automatic scores – except for the handcrafted SUMTIME-hybrid system. Just as in the previous evaluation, this system was not among the better systems according to the automatic metrics, but was given the highest scores from the human evaluators. However, we did not see the same strong performance of the PBSMT-unstructured system on the automatic measures, as our results showed it to be similarly ranked by both the human and automatic metrics.

As for MOUNTAIN, these results show it to be significantly better than the other SMT-based approaches, according to all of the metrics. It is not immediately clear why MOUNTAIN is so much better. Further, it is comparable to or better than the original corpus according to the human raters, which is a positive but somewhat puzzling result. Our best hypothesis for this is that MOUNTAIN regularizes the output from the corpus, making its forecasts' language similar from one to another, while the corpus is written by several different human forecasters who have distinct writing styles and might not get identical ratings. Additionally, the other trained system (PSCFG-semantic) was rated as having the same readability and somewhat better clarity than MOUNTAIN. As we mentioned above, though, that system is not fully automatic like MOUNTAIN, and also makes use of information MOUNTAIN does not; if MOUNTAIN *did* use that information it is likely to improve its output. Finally, though the handcrafted SUMTIME-hybrid system has clearly better output, MOUNTAIN has an even clearer edge in creation effort; the SUMTIME system was reported to take a year to build, whereas MOUNTAIN was able to train a new system in this domain in

System	Clarity	Read.	NIST	BLEU	METEOR
SUMTIME	4.763	4.868	5.317	.5548	.6840
PSCFG-sem	4.723	4.644	5.795	.5876	.7967
MOUNTAIN	4.473	4.631	5.672	.5860	.7784
corpus	4.342	4.513	7.410	1.000	1.000
SMT-unstr	3.763	4.144	4.872	.4619	.4834
SMT-struct	3.171	3.276	3.441	.3088	.1623

Table 2: Mean human-based and automatic scores for 5 NLG systems and the original human-written forecasts. All systems were tested with an identical 12 dates of forecast texts.

	NIST	BLEU	METEOR
Clarity	.926	.957	.953
Readability	.944	.966	.951

Table 3: Pearson's correlation for human-based clarity and readability scores versus the automatic measures.

about a week.

Our results in this evaluation showed high correlation between the automatic measures and the human-based measures. Pearson's $r > 0.9$ for all comparisons, with BLEU and METEOR being generally more highly correlated than NIST for both clarity and readability. The correlation coefficients for each comparison are shown in Table 3.

However, it is not clear that the automatic metrics are designed to measure the same thing that the human evaluation did. Though clarity and readability are important when considering NLG quality, it seems that the automatic measures are actually measuring *adequacy*: how well a particular example says what a reference does. In fact, this is something that has been suggested by results of comparing different evaluation metrics [14] – the automatic measures are more highly correlated to NLG output adequacy than other dimensions. Adequacy is somewhat more challenging to test well in a human evaluation, though. The best method is to show the original data used to generate an example (in the METEO case, that would be the raw input) as well as a machine-generated text, and ask the human to rate how well the text communicates the information from the input. The drawback is that it often requires some expertise to be able to tell if an output sentence is reasonable for a given input. The alternative is to present a human-written example instead of the input data, which requires less expertise for evaluators but can bias the perception of the machine-generated output.

5. Discussion

The earlier large-scale evaluation in this domain [12] examined four general types of systems (handcrafted rule-based, probabilistic context-free grammars, probabilistic synchronous context-free grammars, and phrase-based statistical machine translation), with some types having multiple implementation variants. Of these, the PBSMT approaches were generally among the poorer systems according to both automatic and human-based measures. Here, our results show MOUNTAIN, also a PBSMT-based system, can have performance remarkably close to the handcrafted and PSCFG systems, though still not quite as good. However, the MOUNTAIN approach is fully automatic, whereas the other systems included some amount of manually-encoded linguistic knowledge to achieve their output quality. The fully automatic approach is appealing due to its lower cost of implementation and maintenance. MOUNTAIN clearly outperforms the other PBSMT systems we tested. The most likely difference is in the design and tokenization of the internal language; the other systems used "simply the augmented corpus input vectors", and also tried tagging the vectors with structure information. With the expected use of MOUNTAIN, the system designer has the ability to control the structure and vocabulary of the internal language, which corresponds to underlying structural information (like dialog state). There are likely to be multiple valid and reasonable representations of that information, but not all of them are guaranteed to be equally suitable as input for statistical machine translation. Optimally specifying an input language for a given application is likely to be a challenging problem; finding an automatic method that can produce reasonable options would help in getting the most out of this language generation approach.

Compared to other NLG systems, MOUNTAIN requires relatively little time to set up. The largest and most expensive part is corpus collection; once the training corpus is available, the training time itself is minimal. There are, however, potential solutions for obtaining a suitable corpus without excessive cost. Besides including responses from the system developers, which is similar in cost and skill to template-writing, other data sources such as transcribed Wizard-of-Oz interactions could be used. Potentially, any available human-human dialogs for the application domain could also be included in the training corpus, as long as they could be transcribed and annotated with the internal language.

Finally, though MOUNTAIN appears to work reasonably well in the offline tests that have been done, it has not yet been tried in a full dialog application. While there is plenty of reason to believe MOUNTAIN can be successfully used as part of a spoken dialog application, it is important to verify this within an actual system, as well as to compare its cost and performance to more typical template-based dialog NLG. We are currently in the planning stages of such a test.

6. References

[1] O. Lemon and O. Pietquin, "Machine learning for spoken dialogue systems," in *Interspeech 2007*, Antwerp, Belgium, 2007.

[2] H. Horacek, "Text generation methods for dialog systems," in *2003 AAAI Spring Symposium*, Palo Alto, CA, 2003, pp. 52–54.

[3] Y. W. Wong and R. J. Mooney, "Generation by inverting a semantic parser that uses statistical machine translation," in *NAACL-HLT 2007*, Rochester, NY, 2007, pp. 172–179.

[4] I. Langkilde and K. Knight, "Generation that exploits corpus-based statistical knowledge," in *ACL/ICCL 1998*, Montreal, Quebec, Canada, 1998, pp. 704–710.

[5] A. Ratnaparkhi, "Trainable methods for surface natural language generation," in *NAACL 2000*, Seattle, WA, 2000, pp. 194–201.

[6] T. Marciniak and M. Strube, "Using an annotated corpus as a knowledge source for language generation," in *Workshop on Using Corpora for NLG*, Birmingham, UK, 2005.

[7] M. Banko, V. O. Mittal, and M. J. Witbrock, "Headline generation based on statistical translation," in *ACL 2000*, Hong Kong, 2000.

[8] B. Langner and A. Black, "MOUNTAIN: A translation-based approach to natural language generation for dialog systems," in *IWSDS 2009*, Irsee, Germany, 2009.

[9] S. Sripada, E. Reiter, J. Hunter, and J. Yu, "SUMTIME-METEO: Parallel corpus of naturally occurring forecast texts and weather data," Computing Science Department, University of Aberdeen, Aberdeen, Scotland, Tech. Rep. AUCS/TR0201, 2002.

[10] S. Sripada, E. Reiter, J. Hunter, and J. Yu, "Exploiting a parallel text-data corpus," in *Corpus Linguistics*, Lancaster, UK, 2003.

[11] A. Belz, "Statistical generation: Three methods compared and evaluated," in *ENLG 2005*, Aberdeen, Scotland, 2005, pp. 15–23.

[12] A. Belz and E. Kow, "System building cost vs. output quality in data-to-text generation," in *ENLG 2009*, Athens, Greece, 2009.

[13] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, Ann Arbor, MI, 2005.

[14] A. Stent, M. Marge, and M. Singhai, "Evaluating evaluation methods for generation in the presence of variation," in *CICLing 2005*, Mexico City, Mexico, 2005.

[15] E. Reiter and A. Belz, "An investigation into the validity of some metrics for automatically evaluating natural language generation systems," *Comp. Ling.*, vol. 35, no. 4, pp. 529–558, 2009.

[16] A. Belz and E. Reiter, "Comparing automatic and human evaluation of NLG systems," in *EACL 2006*, Trento, Italy, 2006.