From dictations to clinical reports using machine translation

Greg P. Finley, Wael Salloum, Najmeh Sadoughi, Erik Edwards, Amanda Robinson, Mark Miller, David Suendermann-Oeft*
EMR.AI Inc.
San Francisco, CA, USA
greg.finley@emr.ai

Michael Brenndoerfer
University of California
Berkeley, CA, USA

Nico Axtmann
DHBW, Karlsruhe, Germany

Abstract

A typical workflow to document clinical encounters entails dictating a summary, running speech recognition, and post-processing the resulting text into a formatted letter. Post-processing entails a host of transformations including punctuation restoration, truecasing, marking sections and headers, converting dates and numerical expressions, parsing lists, etc. In conventional implementations, most of these tasks are accomplished by individual modules. We introduce a novel holistic approach to post-processing that relies on machine callytranslation. We show how this technique outperforms an alternative conventional system—even learning to correct speech recognition errors during post-processing—while being much simpler to maintain.

1 Introduction

Medical dictation is one of the most common ways of documenting clinical encounters (Rosenbloom et al., 2010). The dictated material needs to be transformed into a textual representation to be printed as a clinical letter or inserted into electronic medical record (EMR) systems. This can be done using one of the following techniques (Alapetite et al., 2009):

1) the speech recording is manually transcribed by a third party and returned to the physician for sign-off at a later point in time;

2) the recording is processed by a medical speech recognizer, controlled and corrected by a quality assurance team (mostly an external entity), and returned to the physician;

3) while the physician is dictating, a medical speech recognizer transforms the speech into text which is subject to immediate correction and sign-off by the physician.

*Patent pending.

```
this is doctor mike miller dictating
a maximum medical improvement slash
impairment rating evaluation for
john j o h n doe d o e social one
two three four five six seven eight
nine service i_d one two three four
five six seven eight nine service
date august eight two thousand
and seventeen subjective and
treatment to date the examinee is
a thirty nine year old golf course
maintenance worker with the apache
harding park who was injured on
eight seven two thousand seventeen
```

Figure 1: Raw output of a medical speech recognizer.

In this paper, we focus on the text processing that follows the application of automated speech recognition (ASR) in Techniques 2 and 3. The role of ASR is simply to transform spoken words into plain text, as exemplified in the excerpt of a medical dictation in Figure 1: ASR output is typi case insensitive and contains only alphabetic characters, transcribed verbatim including command words, repetitions, grammatical errors, etc.

In contrast, clinical letters follow rigorous formatting standards which require a sophisticated post-processor to transform the ASR output into a full-fledged letter. Major responsibilities of the post-processor include: truecasing, punctuation restoration, carrying out dictated commands (e.g., 'new paragraph', 'scratch that'), converting numerical and temporal expressions, formatting acronyms and abbreviations, numbering itemized lists, separating sections and section headers, and inserting physician "normals" (sections of boilerplate text or templates).

Figure 2 shows a post-processed version of the raw ASR output of Figure 1. This example makes

Proceedings of NAACL-HLT 2018, pages 121–128
New Orleans, Louisiana, June 1 - 6, 2018.

clear that many of the tokens of the ASR output need to be altered in order to create a properly formatted output document. In fact, informal experiments indicated that, on average, *more than half* of spoken tokens are subject to modification when preparing a clinical report from a dictation.

Conventional implementations of post-processors comprise a multitude of predominantly rule-based techniques (Sistrom et al., 2001; Liu et al., 2006; Frankel and Santisteban, 2011), mostly covering subsets of the operations listed above. There has been a fair amount of machine learning research on punctuation restoration (PR), which does constitute a significant component of post-processing, over the last two decades (Beeferman et al., 1998; Peitz et al., 2011). PR has even been addressed for medical ASR specifically, using methods such as finite state models for punctuation insertion (Deoras and Fritsch, 2008), or identifying punctuated tokens using recurrent neural networks (Salloum et al., 2017b).

Of course, PR is only one necessary module of a post-processing system. Typical modular approaches, especially those that are predominantly rule based, are subject to serious disadvantages in practical use. For one, the task may grow in complexity over time through the introduction of specific rules for certain hospitals or physicians. Another issue is that these systems must follow an ASR stage, where unforeseen errors (Johnson et al., 2014; Hodgson and Coiera, 2016; Edwards et al., 2017) may interfere destructively with post-processing, for which rules or models are typically designed or trained for idealized transcriptions.

In this paper, we present a holistic, data-driven approach to post-processing which makes use of recent advances in statistical machine translation, covering most of the aforementioned operations in a single shot and exhibiting accuracy superior to an existing modular system. After a brief introduction to machine translation in Section 2, we describe methods, data sets, and evaluation in Section 3 and experimental results in Section 4.

2 Machine translation

We approach the post-processing problem as a case of machine translation (MT), in which the source language is the raw ASR output as in Figure 1, and the target language is the final written letter from Figure 2—or, more accurately, a

This is Dr Mike Miller dictating a Maximum Medical Improvement/Impairment Rating Evaluation for John Doe.
SSN: 123-45-6789
Service ID: 123 456 789
Service Date: 08/08/17

Subjective and Treatment:
To date, the examinee is a 39 year-old golf course maintenance worker with the Apache Harding Park who was injured on 08/07/17.

Figure 2: Output of post-processor.

form that can be trivially converted into the final text, and in which formatting elements are represented themselves as words. To our knowledge, ours is the first system to frame ASR post-processing as MT, and one of very few described post-processing systems for the medical domain.

Most standard MT approaches require both parallel data (bitexts) and an additional quantity of data in the target language only, which is used to build a language model; likelihoods from both the translational and the language model are balanced during translation (Ney et al., 2000; Koehn et al., 2003, 2007; Lopez, 2008).

Koehn et al. (2003) introduce a phrase-based *statistical machine translation* (SMT) approach. Their model is defined based on Bayes' rule and includes the phrase translation probability, the language model for the target, and the distortion probability of the target language to account for occurrences of reordering. The phrase translation model and the distortion probabilities are trained on the aligned phrases of source and target language, and the language model is trained on the target language. During decoding, a sequence of translated phrases is chosen by performing a beam search to limit the set of phrase candidates.

3 Methods

In this section we describe our methods for preparing data for training, tuning, and evaluating our MT methods. To reframe the post-processing problem as MT is not a trivial matter—it requires careful attention to how training data is prepared, due to the requirements for MT and the peculiarities of medical text; we describe our methods for doing so in 3.2. Crucially, we also explore the integration of our models within a working production

Set	# Reports	# Words
Training	8,775	4,785,986 (Rep.) 5,363,580 (Tra.) 5,681,630 (Hyp.)
Tuning	500	276,551 (Rep.) 311,538 (Tra.) 305,672 (Hyp.)
Development	300	187,472 (Rep.) 211,740 (Tra.) 209,587 (Hyp.)
Test	300	177,756 (Rep.) 198,722 (Tra.) 196,198 (Hyp.)

Table 1: Statistics of the data sets used for training, tuning, development, and test.

system in 3.4, enabling us to evaluate the contribution of MT towards improving real-world results.

3.1 Data sources

All models and experiments in this paper use actual clinical notes. Reports and dictations from a variety of specialties at two different US hospitals were considered. As required under HIPAA, EMR.AI has a Business Associate Agreement with the Covered Entity that supplied the data.

We first identified a set of 9,875 reports for which we had manual transcriptions and ASR hypotheses available. This set was split into four smaller sets (see Table 3.1 for corpus statistics). The training set was used to generate source-to-target alignments and build the phrase and distortion models, as well as to train the monolingual language model. (The latter was trained on additional text as well, for a total of 23,754 reports and 14,208,546 words.) The tuning set was used for tuning the relative contribution of the various models for MT. The development set was used for evaluation along the way. Finally, we set aside a blind test set, used solely and exclusively for the results in this paper.

We also set out to test whether transcriptions or hypotheses make better training data. As the task is posed, hypotheses would seem more relevant; however, they are a noisier source of data than transcriptions, and it was not guaranteed that the needed correspondences could be learned through the noise. Therefore, for both training and tuning, we tried transcripts, hypotheses, or a combination of the two (nine separate conditions).

3.2 Finding parallel training samples

Although our data set contains dictations and their corresponding reports, these do not represent true bitexts of the type that are typically used for MT, for several reasons: boilerplate language or metadata may be added to the letter; whole sections may be reordered, or even inserted from prior notes in the patient's history; pleasantries, discontinuities, or corrections by the speaker will be omitted. Furthermore, notes can be thousands of words in length, and it is not practical to learn alignments from such long "sentences" given computational constraints.

To solve these problems, we developed a method to extract matching stretches of up to 100 words from the source and target, which can then be used as training samples. The procedure entails five major steps.

Text preprocessing. Punctuation, newlines, tabs, headings, and list items are separated from adjacent tokens and converted into dummy tokens. All digits become their own tokens.

Dynamic alignment. All matches and edits between source and target are determined using a dynamic program, similar to that used for Levenshtein distance but with key differences: matches are permitted between non-exact string matches if they are determined, in a previous run of the algorithm, to be possible substitutions; edits can be longer than one token; and extending an edit incurs a lesser penalty than beginning a new edit.

Merging edits. Short substitutions are merged together if there is an intervening single-word match between them, and the entire range is considered a substitution. The resulting edits allow for longer stretches of parallel sentence data.

Calculating confidence. For every edit, a score is calculated based on a mix of statistics (calculated from a prior run of the dynamic program), and a heuristic that assigns higher scores to longer substitutions, to shorter insertions or deletions, and to edits that are adjacent to other long edits.

Extracting sentences. An iterative algorithm traverses all edits and matches from left to right, building a parallel source–target "sentence" as it goes. A sentence ends when an edit of too low confidence is reached, or once it exceeds 100 words. In the latter case, the next sentence will start one-third of the way through the previous one, so sentences may overlap by up to 67 tokens.

Each extracted sentence becomes a training

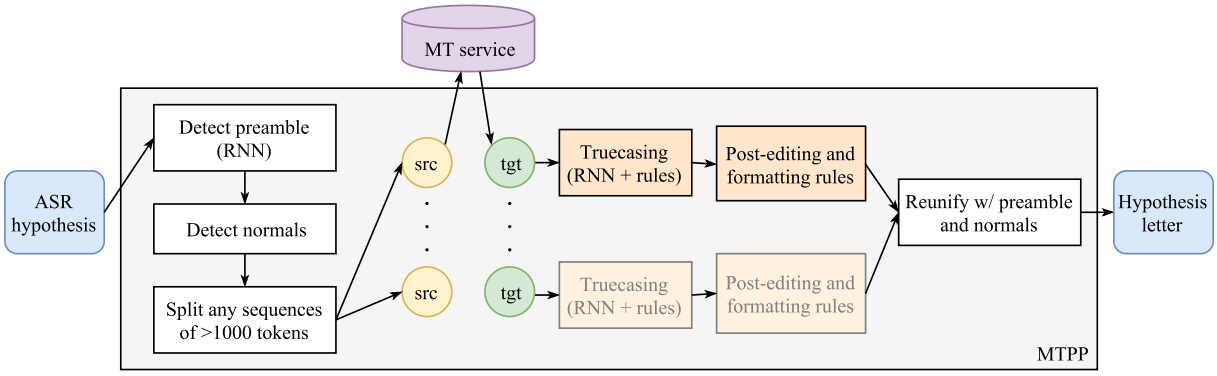

Figure 3: Basic design of the MTPP. Stages within the MTPP shaded in orange are responsible for transforming the MT target language ("tgt") into properly formatted written language.

sample. Any single-word string matches are also written as training samples—because this is not a typical MT problem in that the source and target "languages" are both English, we want to bias the system towards simply regurgitating input words it does not know how to translate. From 8,775 reports, this method generates many training samples: 4,402,612 for transcripts, 4,385,545 for hypotheses, and 8,788,157 for the combined set.

3.3 Model training

For the translation model, we employed typical statistical MT techniques. Optimal word-to-word alignments between source and target were learned using expectation maximization (Och and Ney, 2003). Subject to these alignments, parallel phrases of up to seven words in length were extracted. For the monolingual language model, we trained a 6-gram model with typical interpolation and backoff parameters.

The MT training stage yields a phrase substitution model and a distortion model. To determine the relative contribution of the phrase, distortion, and language models in computing translation option likelihoods, we tuned using minimum error rate training (Och, 2003): translate all text in a held-out tuning set, iteratively adjusting the weights of each contributing model until convergence on an error metric. We used an interpolation of word error rate (WER) and CDER (Leusch et al., 2006), which only assesses a single penalty for "block" movements. We include CDER to reduce the impact on tuning when entire sentences are reordered between the dictation and final letter; note that WER would assess numerous single-word insertion and deletion penalties in such a case.

3.4 Integration with medical post-processor

To use the MT system in production, it had to be integrated into a complete software product, which we refer to as the machine translation post-processor (MTPP), responsible for all stages of transformation between the raw ASR hypothesis and the generated report. Although the bulk of the decisions made during this process are handled by MT, the MTPP is responsible for selecting and preparing inputs for MT and transforming outputs into human-readable form. We present a simplified graphical overview of the MTPP in Figure 3.1.

At the first stage, the "preamble" (spoken metadata that is often not present in the final report) and any commands to insert a template are isolated and not sent to MT. Of the pieces that are subject to MT, any that exceed 1,000 tokens are split. The resulting chunks are sent to an MT daemon which has models pre-loaded into memory and can perform multiple translations in parallel. To each translated chunk, we apply truecasing and post-editing: several steps including joining digits, formatting headings, counting and labeling entries of numbered lists, etc. Finally, all chunks are unified and put into the correct order.

The preamble detector is based on a two-class recurrent neural network (RNN) classifier with pre-trained word embeddings and long short-term memory (LSTM) units, which tags tokens as either in- or out-of-preamble, then finds the split boundary according to a heuristic. The RNN truecaser has a similar architecture but predicts one of three classes for each token—all lowercase, first letter uppercase, or all uppercase—through one layer of softmax output shared across all time frames. This classifier was trained on automatically generated data from 15,635 reports. Truecasing is also sup-

ported through rule-based decisions as well as lists of truecased forms compiled from ontologies and prior reports, which include non-initial capitalizations ('pH', e.g.).

3.5 Evaluation

We assess performance of all models in two text domains: the *MT target domain*, which is the text format described in Section 3.2 in which numerals are split into individual digits, headers are surrounded by dummy tokens, and case is ignored; and the *post-processor error rate (PER) domain*. The latter is used to estimate the manual effort required to correct errors in the hypothesis report. PER can only be calculated from final outputs of a post-processor, and thus depends upon the integration described in Section 3.4.

PER is calculated similarly to WER except that it considers punctuation, newlines, and tabs as separate tokens, and it excludes any detected preamble from consideration (keeping the preamble leads to a slight increase in PER globally). PER is an especially harsh metric in real-world use, as it penalizes ASR errors, post-processing errors, and any other source of distance between the post-processor's output and the final letter following multiple rounds of manual review.

We measure PER of the MTPP against a baseline system, which was also developed internally within EMR.AI for specific use with clinical dictations. The baseline system employs a modular pipeline, where each module is responsible for a particular transformation—for instance, one detects a metadata-heavy "preamble" in the dictation (Salloum et al., 2017a); another converts spelled-out numbers to numerals, dates, etc. Some components of the system are rule based, while others rely on machine learning. This system had been the focus of significant development previously and was in regular production use prior to the advent of the MTPP.

4 Results

In the MT target domain, we present three standard measures of MT performance: WER, CDER, and BLEU. Results for all possible configurations of training and tuning data sources are given in Table 2. Note that these results are on a filtered test set: only source texts of 1,000 tokens or fewer were used (190 out of 300 in the test set), as this was found to be a point beyond which decoding slowed considerably. Note that all BLEU are well above 0.7; these may appear to be exceptionally high scores, but note that our task here is *easier* than a "standard" translation task—to give some idea of a baseline, comparing the totally untranslated dictations in the test set to their matching reports yields a BLEU of 0.318 (as well as WER 0.514, CDER 0.483), which would be quite impossible in a case of translating between two different languages.

For the realistic evaluation of the complete system, we present PER measurements on final outputs of the MTPP in Table 3. Because the MTPP contains logic for breaking up the translation task across longer notes, no filtering is necessary and all 300 notes in the test set can be used. We must emphasize that these results cannot be compared with any quantities in Table 2, as they are measured in different domains entirely.

Tune \ Train	Hyp.	Tra.	Hyp. + Tra.	Metric
Hyp.	0.742	0.746	0.741	BLEU
	0.266	0.277	0.262	WER
	0.170	0.171	0.170	CDER
Tra.	**0.754**	0.745	0.747	BLEU
	0.259	0.276	**0.258**	WER
	0.164	0.171	0.167	CDER
Hyp. + Tra.	0.751	0.721	0.748	BLEU
	0.273	0.317	0.262	WER
	0.166	0.167	0.166	CDER

Table 2: Evaluation of test set on different training and tuning configurations with BLEU, WER, and CDER.

Tune \ Train	Hyp.	Tra.	Hyp. + Tra.
Hyp.	0.322	0.331	0.324
Tra.	0.324	0.338	**0.321**
Hyp. + Tra.	0.328	0.349	0.323

Table 3: Evaluation of the test set on different training and tuning configurations in terms of PER.

Method	PER	
	In: hyp.	In: tra.
No post-processing	0.619	0.574
Non-MT post-proc.	0.411	0.341
MTPP (best MT model)	**0.321**	**0.271**

Table 4: Comparison of PER in several conditions. Results are reported using ASR hypotheses as input ("In: hyp."), as in our other experiments, as well as using manual transcriptions as input ("In: tra.").

Tra.	... her mother was here and had them gave her **ibuprofen** as soon as she started ...
Hyp.	... her mother was here and have him give her **an i_v profile** missing she started ...
MTPP	... her mother was here and gave her **ibuprofen** missing, she started ...
Tra.	in the meantime comma i will have **hospitalist come by and see** the patient ...
Hyp.	in the meantime comma i will have **our hospital was combine to** the patient ...
MTPP	In the meantime, I will have **hospitalist was come by and see** the patient ...
Tra.	carafate one gram **a_c and h_s venlafaxine e_r** seventy five milligrams a day
Hyp.	carafate one gram **a_c_n_h_s meloxicam m e r** seventy five milligrams a day
MTPP	6. Carafate 1 g **before meals and at bedtime. / 7. Venlafaxine ER** 75 mg a day.

Table 5: Examples where the MTPP has "corrected" ASR errors. In each set of three lines, the first is the manual speech transcript, the second is the ASR hypothesis of the same audio, and the third is the output of the MTPP given the ASR hypothesis. Bolded text shows where the MTPP has generated output closer to the actual speech than to its input. Note, for the third example, that the abbreviation 'a.c.' (*ante cibum*) indicates to take the medication before meals, and 'h.s.' (*hora somni*) at bedtime.

The comparison of PER between all nine conditions suggests that the best results are achieved on training data that includes ASR hypotheses (test of proportions: $\chi^2 = 533$, $p < .001$, when comparing average PER with and without hypotheses in training). This is not a highly surprising result, as the evaluation task is to translate hypotheses, although we had wondered before if hypotheses were too noisy to constitute good training data. For tuning data, it appears that either hypotheses or transcripts yield good results, but a mixed set is always worse ($\chi^2 = 44.8$, $p < .001$, comparing average PER when tuned on the mix to PER when tuned on transcripts).

To quantify the impact of MT on post-processing accuracy, we also measured PER of the source hypotheses both before any post-processing and after passing through our baseline post-processor. Results are reported in Table 4. Overall, the MTPP results in a significant decrease in PER from the previous post-processor: a relative reduction of 21.9% error rate for hypotheses ($\chi^2 = 4102$, $p < .001$).

For further context, we also report PER using manual speech transcriptions as input (the rightmost column of Table 4). This is not a realistic use case, but we provide the measurements here to give a sense of the effect ASR errors have on typical PER measurements. The ASR WER of our MT test set was 0.142—much greater than the observed PER difference between hypotheses and transcripts, indicating that many formatting errors in PER occur on the same tokens as ASR errors.

4.1 Correcting ASR mistakes

For the MT models that learn from hypotheses, it was conceivable that they could actually learn to correct ASR mistakes by identifying common error patterns and how they are typically corrected in the final letter. To the MT system, there is no essential difference between, say, inserting formatting elements around a section header and replacing an erroneously recognized phrase with the intended phrase from the report; all words, numerals, and structural elements are tokens alike.

Indeed, we found several occurrences in our test set of phrases in MTPP output that were more similar to manual transcriptions of these dictations than to the ASR hypotheses that served as input to the MTPP. Refer to the examples in Table 5: each shows a transcript of a segment of speech (first line), the ASR hypothesis on that same segment (second line), and the output of the MTPP when given the ASR hypothesis as input (third line). In each, the MTPP output contains a bolded segment that is closer to the transcription than to the hypothesis. (Although note some incomplete cases, such as "hospitalist was come by and see" in the second example.) All of these examples were taken from the same test set as the other results in this paper. None of the transcriptions from the test set were ever seen by any system during training, tuning, or testing (all previous quantitative results used ASR hypotheses, not manual transcriptions, as the source language).

5 Discussion

Using MT for the post-processing task has numerous advantages over other approaches. Most ob-

viously from our results, it achieves a high level of accuracy, even roundly outperforming a system containing numerous hand-designed rules and deep learning approaches that were trained on large amounts of annotated data.

Additionally, MT is a better solution for an adaptable and improvable system. The core of the system can be adapted to other dialects of English or even other languages by retraining the models. Even in the simplest use case, however, retraining can be periodically undertaken to improve performance on current data, accounting for possible changes over time in dictation or report writing style, as well as any ongoing development of the associated speech recognizer.

A final advantage is in the cost of maintaining the system. Although MT training has relatively high compute and memory requirements, there is very little cost in human time to retrain new models. Although our very best results did use transcriptions, our experiments demonstrate that the entire process can be reproduced fruitfully without them (and may even be subject to less unpredictability). To continuously improve a rule-based system, direct human intervention is required to write and validate new rules. For any supervised machine learning modules of a post-processor, human annotators may also be required.

6 Conclusion

In this paper we presented an overview of a complete and validated medical ASR post-processing system that relies on MT, as well as the novel processing methods required to ensure that MT is a viable approach for clinical dictations. Our strategy has multiple significant advantages compared to traditional rule-based approaches, and even other machine learning-based ones—not only does the MT design result in substantially reduced formatting errors, achieved in part by its ability to correct errors made by the speech recognizer in the first place, but it can also be retrained and improved fully automatically, without the need for costly manual adjustments.

References

Alexandre Alapetite, Henning B. Andersen, and Morten Hertzum. 2009. Acceptance of speech recognition by physicians: a survey of expectations, experiences, and social influence. *International journal of human-computer studies*, 67(1):36–49.

Doug Beeferman, Adam Berger, and John D. Lafferty. 1998. Cyberpunc: a lightweight punctuation annotation system for speech. In *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 689–692. IEEE.

Anop Deoras and Juergen Fritsch. 2008. Decoding-time prediction of non-verbalized punctuation. In *Proc. of Interspeech*, pages 1449–1452, Brisbane, Australia. ISCA.

Erik Edwards, Wael Salloum, Greg P. Finley, James Fone, Greg Cardiff, Mark Miller, and David Suendermann-Oeft. 2017. Medical speech recognition: reaching parity with humans. In *Proc. of SPECOM*, volume LNCS 10458, pages 512–524. Springer.

Alan Frankel and Ana Santisteban. 2011. System and method for post processing speech recognition output. US Patent 7,996,223.

Tobias Hodgson and Enrico W. Coiera. 2016. Risks and benefits of speech recognition for clinical documentation: a systematic review. *Journal of the American Medical Informatics Association*, 23(e1):169–179.

Maree Johnson, Samuel Lapkin, Vanessa Long, Paula Sanchez, Hanna Suominen, Jim Basilakis, and Linda Dawson. 2014. A systematic review of speech recognition technology in health care. *BMC medical informatics and decision making*, 14(94):1–14.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of the 45th annual meeting of the ACL*, pages 177–180. ACL.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of the Conference of the North American Chapter of the ACL on Human Language Technology*, volume 1, pages 48–54, Edmonton, Canada. ACL.

Gregor Leusch, Nicola Ueffing, and Hermann Ney. 2006. Cder: Efficient mt evaluation using block movements. In *Proc. of the 11th Conference of the European Chapter of the ACL*, pages 241–248, Trento, Italy. ACL.

David Liu, Mark Zucherman, and William B. Tulloss. 2006. Six characteristics of effective structured reporting and the inevitable integration with speech recognition. *Journal of digital imaging*, 19(1):98–104.

Adam Lopez. 2008. Statistical machine translation. *ACM computing surveys*, 40(3, article 8):1–49.

Hermann Ney, Sonja Niessen, Franz J. Och, Hassan Sawaf, Christoph Tillmann, and Stephan Vogel. 2000. Algorithms for statistical translation of spoken language. *IEEE transactions on speech and audio processing*, 8(1):24–36.

Franz J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41st Annual Meeting of the ACL*, pages 160–167. ACL.

Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Stephan Peitz, Markus Freitag, Arne Mauser, and Hermann Ney. 2011. Modeling punctuation prediction as machine translation. In *Proc. of the International Workshop on Spoken Language Translation (IWSLT)*, pages 238–245. ISCA.

S. Trent Rosenbloom, William W. Stead, Joshua C. Denny, Dario Giuse, Nancy M. Lorenzi, Steven H. Brown, and Kevin B. Johnson. 2010. Generating clinical notes for electronic health record systems. *Applied clinical informatics*, 1(3):232–243.

Wael Salloum, Greg Finley, Erik Edwards, Mark Miller, and David Suendermann-Oeft. 2017a. Automated preamble detection in dictated medical reports. *BioNLP 2017*, pages 287–295.

Wael Salloum, Greg P. Finley, Erik Edwards, Mark Miller, and David Suendermann-Oeft. 2017b. Deep learning for punctuation restoration in medical reports. *Proc. of the 55th Annual Meeting of the ACL, Workshop on Biomedical Natural Language Processing (BioNLP)*, pages 159–164.

Chris L. Sistrom, Janice C. Honeyman, Anthony Mancuso, and Ronald G. Quisling. 2001. Managing predefined templates and macros for a departmental speech recognition system using common software. *Journal of digital imaging*, 14(3):131–141.