The University of Melbourne

COMP90090 - Natural Language Processing for Health

Semester 1, 2023

Assignment 3: Independent Project

Optimization and deployment challenges of closed-source LLMs for clinical note abbreviation expansion

[redacted]

ABSTRACT

Objective: To explore optimization and deployment challenges of closed-source large language models in clinical note abbreviation expansion.

Methods: We tested a range of prompts on the clinical abbreviation expansion dataset, CASI, with 3 models from OpenAI's GPT-3.5 range. We refined prompts on a 'hard' subset of the data before scaling six model-prompt combinations to 1.9k instances, and the best model-prompt combination to a further 18.2k instances. We evaluated performance on each shortform and compared macro F1 and average accuracy against CASI benchmarks.

Results: There was a baseline performance leap to each newer OpenAI model. Prompt refinement further increased performance. Our best model-prompt combination matched state-of-the-art performance in accuracy on the CASI dataset, albeit with lower F1 score. The models exhibit erratic behavior and show strong indications that the CASI dataset were included in the model training data.

Discussion: The closed-source models exhibit strong overall abbreviation expansion performance. However, they exhibit variant performance across each of the 41 shortforms and idiosyncratic variance with prompts on each model. There are strong indications that the CASI dataset was part of the training data, suggesting that performance metrics may not be replicated on unseen data. There are also confidentiality concerns using closed-source models via an external API call which may limit their applicability in clinical applications.

Conclusion: Closed-source LLM's exhibit exceptional performance in information extraction from clinical notes. However, confidence in the deployment of this tool in production in a clinical environment will be complicated be likely concerns of performance generalizability to private data, erratic behavior and confidentiality concerns.

INTRODUCTION

Clinical abbreviation expansion

Abbreviations and medical jargon are common in clinical notes. Acronyms can refer to one of many expansions. 'PT', for example, can mean 'physical therapy' in one context or 'posterior tibial' in the next. This ambiguity leads to imprecision in downstream tasks which are dependent on the information extracted from clinical notes, including point of care services and clinical research.

The importance and difficulty of abbreviation expansion has been understood for several decades.[1]

Attempts to automate the task have progressed over these years.[2–4] Potential solutions have been built on rules-based algorithms which match text patterns,[5] sense inventories[6] and supervised machine learning.[4] Notable performance gains were obtained using contextual-embedding approaches from the late-2010s.[7,8]

Recently, general-purpose, closed-source large language models have shown exceptional performance on many information extraction tasks,[9] including clinical information extraction.[10]

LLMs use prompts

Rather than fine-tuning for a specific task, language tasks are solved on large language models using prompts. A prompt is a text string which the language model probabilistically completes so that the output look like the strings it solved during its training phase.[11] Prompting can be thought of as a method to locate within a large language model a task that has already been learned by it.[12] Because prompts are the only way to guide these models to produce the desired output, a dedicated field of research into 'prompt engineering' has emerged,[13–15] including in healthcare.[16]

A word of caution

Despite the rise in the popularity of these models, research suggests their use should be accompanied by caution. Small changes in prompt design have been shown to vary answer content and correctness.[17–19] Output varies from one model upgrade to the next,[20] and not always in a positive direction.[21] Training data and system architecture cannot be interrogated because they have not been made public on the popular closed-source models, including OpenAI's recent models.[9] Use of these models on sensitive clinical data is also complicated by standard conditions of use which permit storage and monitoring of API content by these companies. OpenAI, for instance, stores prompt content for inspection for 30 days.[22]

Using closed-source LLMs for clinical abbreviation expansion

In this paper we explore the use of closed-source LLMs on the clinical abbreviation expansion problem. We propose a methodology to refine and optimize prompts for this task.

We also recognize the cautionary notes raised about the use of these models. We therefore consider not only performance optimization, but also the other challenges of deployment in the clinical context including output variability and patient confidentiality.

METHODS

Full details of the software architecture, dataset, models and prompting techniques are set out in **Appendix A**.

We conducted several experiments on different subsets of the data, progressively expanding the size of the dataset whilst narrowing the range of prompt-model combinations being tested.

Models

We use 3 closed-source models from OpenAI. Details of the models are set out in **Table 1**.

Table 1: Details of models used in this paper

Model name	Details (per OpenAI[23–25])
`text-davinci-002`	Trained with supervised fine tuning on text and code to Jun 2021.
`text-davinci-003`	Trained with reinforcement learning on text and code to Jun 2021. Claimed improvement on text-davinci-002.
`gpt-3.5-turbo`	Trained on text and code to Sep 2021. Claimed Improvement on text-davinci-003 and optimized for chat.

Datasets

We use the Clinical Acronym Sense Inventory (CASI), a publicly available anonymized dataset of clinical acronyms.[26]

Table 2: Example instance from CASI

Field	Data
Shortform:	AV
Longform:	arteriovenous
Context:	CHEST: Clear. HEART: Rhythm is normal. EXTREMITIES: No effusion in his left knee. Has fair range of motion. Has an AV access fistula on his left forearm and has a good bruit over it. LABORATORY DATA: Hemoglobin and potassium will be drawn and faxed to the hospital.

Each instance in the dataset is a clinical note extract containing a shortform, and the corresponding longform. The dataset is derived from clinical notes from various specialties in hospitals affiliated with the University of Minnesota. An example instance is shown in **Table *2***. Several performance benchmarks have been published for CASI, permitting us to conduct a performance comparison.[8,10] Unlike many clinical databases, there is no restrictions on sending the content from CASI to the external OpenAI APIs.

We built several datasets from the full CASI set. These are set out in **Table 3**.

Table 3: Datasets used in this paper

Name	Content	Source	Usage in this paper
Full CASI	37,500 clinical note extracts, containing 75 abbreviations ('shortforms' (sf)) and 351 expansions ('longforms' (lf))	Original CASI set, per Moon et al.[26]	Not used
main_set	18,233 instances, containing 41 sf and 149 lf.	De-noising of the Full CASI set, per Adams et al.[8]	Assess best model-prompt performance.
small_set	1,917 instances, containing 41 sf and 149 lf.	~10% of the Reduced-CASI set, in same longform proportions.	Assess 6x model-prompt performance, to see if prompt improvements scale.
hard_set	200 instances, containing 37 sf and 85 lf.	'Hard' samples from main_set which various GPT-3 and GPT3.5 models failed to correctly expand.	Assess high frequency iterations of model-prompt variations.
rare_set	190 instances, containing 1 sf and 1 lf.	From Full CASI, where longform equals 'Fairview Southdale Hospital' (a rare, proper noun) and the long form does not appear in the context.	Assess whether models predict the rare proper noun, indicating that dataset is within model training data.

Ethics statement

As noted above, the CASI dataset was selected because it is anonymized. The clinical notes have been de-identified by replacing names and dates with placeholders. The dataset only contains a clinical note extract, not any other details of the patient's record. The dataset is expressly made available '*for public use*.'[26] It has previously been used in published studies using an OpenAI API.[10] Therefore, no ethics approval for the study was sought.

Problem structure and software architecture

Each CASI instance contains a shortform, a target longform, and context. For each shortform, there are a number of longform options in the dataset. The goal is to correctly expand the shortform into the longform. Similarly to Agrawal et al, we treat the problem as a multiple choice task in which the model's raw output is 'resolved' into an integer index from the multiple choice list.[10] Details of the resolver and other architecture is set out in **Appendix A**.

Experiments

Experiment 1: Prompt iterations on hard set

Our first goal was to explore the performance impacts of different prompt and model combinations.

The prompt is a concatenated string containing several sub-components. This includes an instruction on the model's task and its substantive payload, namely the clinical note extracts and shortforms to be expanded.

The first instruction we tested was akin to that used by Agrawal et al and was accompanied by the full extract of each instance:[10]

> "Expand the abbreviations in the following sentence snippets. [format instructions]. *(for each instance in the batch):* What does '[shortform]' in the following sentence refer to?: '[full extract]'."

We then iterated this instruction with a series of minor amendments. In total we sampled 14 different

combinations of instructions and payloads. We tested each combination on each of the 3 models over the 200 instances in the `hard_set`. The overall accuracy of inferences was recorded.

The concatenated prompt string also includes an instruction to the model to format its response in a certain manner (referred to above as the '[format instruction]'). This is necessary because the model response is an unstructured string. The response therefore needs to be in a consistent format so that the results can be parsed automatically. We developed a specific instruction that the model return it's answer in a given format. We also developed a corresponding parser. We did not alter these throughout our testing. Details of this formatting instruction are set out in **Appendix A**. All prompts tested are set out in **Appendix B**.

Experiment 2: Scale to small set

Our second goal was to see if the optimal prompts from Experiment 1 would scale to a larger set with more representative shortform distributions. We took both the initial and the best performing prompts from Experiment 1 for each model and ran them over the `small_set`. We obtained accuracy and macro F1 score for each of the 6 tests.

Experiment 3: Scale to main set

Our third goal was to obtain overall performance metrics on the `main_set`. We used the single best performing model-prompt combination from Experiment 2. We recorded accuracy, macro and weighted precision, recall and F1 for each 41 shortforms. A zero-R majority baseline was used for reference.

Experiment 4: Rare long forms

Our final experiment stands alone from the first three. We sought to infer whether the original CASI dataset was part of OpenAI model training data in order to infer whether the performance results in Experiments 1 to 3 were likely to be replicated on unseen data. We ran inferences on the rare dataset containing solely instances of the shortform '*FSH*' and longform '*Fairview Southdale Hospital.*' We manually reviewed the unprocessed model inferences and recorded the number of references to *'Fairview Southdale Hospital'*.

RESULTS

Experiment 1: Prompt iterations on hard set

We sampled 14 combinations of instruction, payload and batch size combinations for each of the three models and recorded the prediction accuracy of each. Full prompt details are found in **Appendix B**. Accuracy results are shown in **Figure 1**.

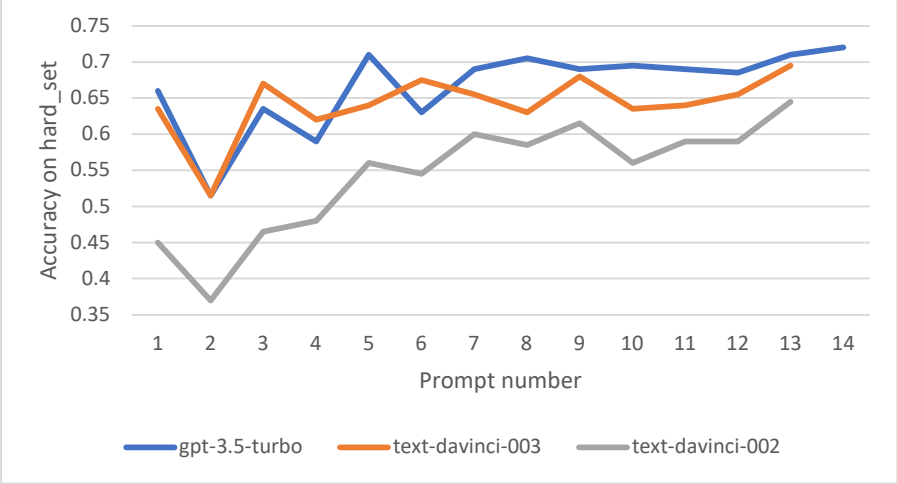


Figure 1: Accuracy of various prompt combinations for the three models on the `hard_set`

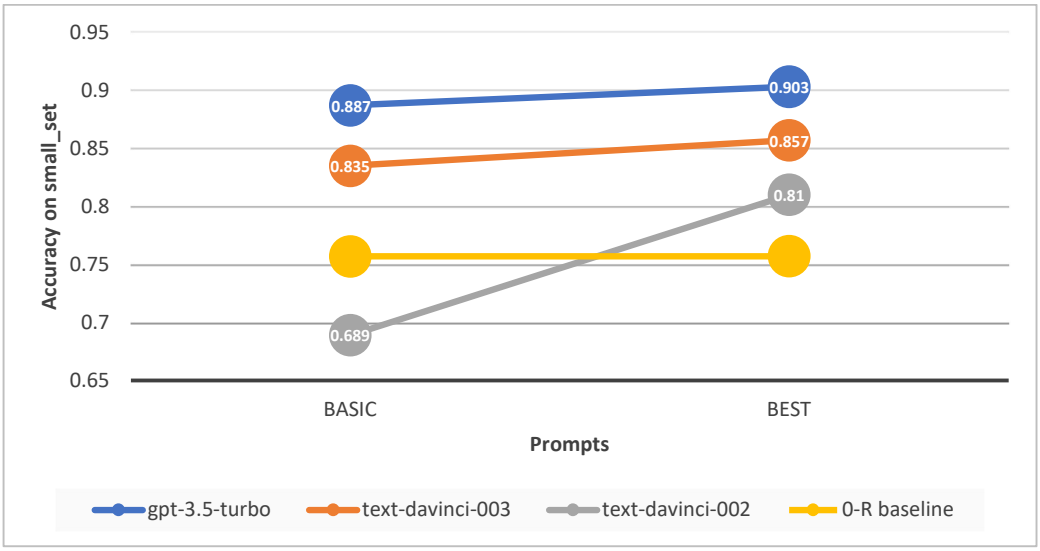


Figure 2: Changes in accuracy over the three models when using a basic prompt and the 'best' prompt on the `small_set`

We found that there was a clear performance leap from `text-davinci-002` to `text-davinci-003`, with a smaller average, but inconsistent, performance gain to `gpt-3.5-turbo`. This is consistent with experiments by Ye et al on other NLP tasks.[21]

We also found that model behavior can be erratic. Prompts which increase performance on one model can decrease performance on another. For example, adding the subheading (ie. the local header in which the shortform is found) to the context snippet increased `gpt-3.5-turbo` performance, but reduced it for `text-davinci-003`.

Subtle prompt changes can also impact performance. Adding the simple instructions *'You are a helpful assistant'* to the beginning of the prompt increased performance on two of the three models, but decreased performance in another.

We found that maximizing the specific details about the immediate shortform context, such as using the unaltered original text (in preference to a preprocessed text with stop words and punctuation removed) and marking the shortform in the text with marker characters (eg '*~~PDA~~*'), tended to improve performance.

Experiment 2: Scale to small set

We used both the initial, and the best, prompts for each of the three models from Experiment 1, and ran them over the `small_set`. Accuracy of each is shown in **Figure 2**.

We found that findings from Experiment 1 were scalable. Prompts and models with the best performance on the `hard_set` also performed best on the `small_set`, and performance jumps using the later models are evident. The prompt and settings with the optimum performance are shown in **Appendix C**.

Experiment 3: Scale to full set

We ran the optimum prompt from Experiment 2 on the `main_set`.

Overall accuracy was 0.905 and macro F1 was 0.687. This matched existing gold standards (set by Agrawal et al using GPT-3[10]) for accuracy, but with a slight decrease in F1. Comparative results are shown in **Table 4**.

Detailed metrics for each shortform are found in **Appendix D**. A small number of shortforms exhibit notably poor performance below two standard deviations from the mean accuracy. For example, the shortform 'C&S' had a baseline 0-R accuracy 0.88, but had prediction accuracy 0.62 and mF1 of 0.34. Examination of the results showed it had strong (yet inexplicable) tendency to predict the shortform as *'culture and sensitivity'* over 500% more frequently than it appears in the dataset.

Table 4: Comparison of benchmark results on CASI, including our results on `gpt-3.5-turbo`

Method	Accuracy	Macro F1
0-R baseline	0.79	0.28
LMC[8]	0.71	0.51
GPT-3 edit-R distillation[10]	0.90	0.76
gpt-3.5-turbo (our study)	0.90	0.69

Experiment 4: Rare long forms

We ran the predictions on the `rare_set` on the `gpt-3.5-turbo` model. The number of instances returned by `gpt-3.5-turbo` with different batching conditions is shown in **Appendix E**. We found that up to 57% of predictions for the acronym *'FSH'* were predicted to be *'Fairview Southdale Hospital.'*

It seems highly unlikely that the model would have predicted a specific location name with that frequency unless the CASI dataset, or some derivation of it, were included in the model's training data. That suggests that, despite the strong performance in Experiment 3, this performance level may not be replicable on new, unseen data.

DISCUSSION

Strengths of closed-source LLMs for clinical NLP tasks

LLM's have garnered significant attention for their generalizable NLP abilities, especially since the recent release of ChatGPT. Much of that enthusiasm seems warranted. A litany of papers show it obtaining performance leaps over previous gold standards in a series of NLP tasks in both clinical and other domains.[10,27] Our experiments show matching high levels of performance on the clinical abbreviation expansion task.

The potential to use these models to extract and structure previously obscured data in clinical notes seems high. If used intelligently in point of care applications and health research, improved health outcomes are likely to follow.

Performance variability

Even though average performance beat pre-LLM gold standards, the models have many shortcomings. As evidenced in this study, they are prone to erratic performance drops triggered by minor adjustments to their prompts. That suggests that any use of the models in a production environment requiring predictability and precision, such as clinical research but particularly real-time clinical operations, needs to be closely constrained and rigorously tested.

Our results in Experiment 4 strongly suggest that the CASI dataset was contained in the model training data. It is therefore likely that model performance on unseen clinical abbreviation data is likely to be lower than indicated in this paper.

The fact that we can only speculate on the training data is an example of a broader concern with using closed-source models for processing new, unseen data, namely that the model's bias, accuracy, tendency to hallucinate and other performance metrics cannot be easily be predicted or interrogated.

Patient confidentiality and external-APIs

The other major concern of using a closed-source model via an external API in the clinical context is that of patient confidentiality. As noted in the introduction to this paper, the CASI dataset was chosen because it does not contain express restrictions on data use and storage which would prevent external API use. However, many clinical datasets, and indeed most patient data in any form, is likely to be covered expressly or implicitly by data use agreements, privacy legislation and/or medical ethics obligations preventing the transmission of the data to an external API.

Recent developments in logging and storage policies at Microsoft and OpenAI may partly address these concerns. Microsoft acknowledge that *"[s]ome customers may want to use the Azure OpenAI Service for a use case that involves the processing of sensitive, highly confidential, or legally-regulated input data but where the likelihood of harmful outputs and/or misuse is low."*[28] Special application can be made to them for private data to be processed in a dedicated Azure cloud instance and to opt out of the human review process in which data is stored by the host for 30 days and can be inspected by their employees. Whilst this may not sufficiently address all confidentiality concerns for sensitive clinical data, it suggests that markets may develop which accommodate use of external APIs for sensitive clinical data as this new field matures.

CONCLUSION

Closed-source LLM's exhibit exceptional performance in information extraction from clinical notes. However, confidence in the deployment of this tool in production in a clinical environment will be complicated be likely concerns of performance generalizability to unseen data, erratic behavior and confidentiality concerns.

REFERENCES

1. Liu, H., Lussier, Y. & Friedman, C. A study of abbreviations in the UMLS. *Proc AMIA Symp* **2001**, 393–39.

2. Pakhomov, S., Pederson, T. & Chute, C. Abbreviation and Acronym Disambiguation in Clinical Discourse. *AMIA Annu Symp Proc. 2005* **2005**, 2005: 589–593.

3. Pakhomov, S. Semi-supervised Maximum Entropy based approach to acronym and abbreviation normalization in medical texts. in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02* 160 (Association for Computational Linguistics, 2001). doi:10.3115/1073083.1073111.

4. Xu, H., Markatou, M., Dimova, R., Liu, H. & Friedman, C. Machine learning and word sense disambiguation in the biomedical domain: design and evaluation issues. *BMC Bioinformatics* **7**, 334 (2006).

5. Schwartz, A. & Hearst, M. A simple algorithm for identifying abbreviation definitions in biomedical text. *Pac Symp Biocomput.* **2003**, 451–62.

6. Xu, H., Stetson, P. D. & Friedman, C. Methods for Building Sense Inventories of Abbreviations in Clinical Notes. *Journal of the American Medical Informatics Association* **16**, 103–108 (2009).

7. Skreta, M. *et al.* Automatically disambiguating medical acronyms with ontology-aware deep learning. *Nat Commun* **12**, 5319 (2021).

8. Adams, G., Ketenci, M., Bhave, S., Perotte, A. & Elhadad, N. Zero-Shot Clinical Acronym Expansion via Latent Meaning Cells. (2020) doi:10.48550/ARXIV.2010.02010.

9. OpenAI. GPT-4 Technical Report. (2023) doi:10.48550/ARXIV.2303.08774.

10. Agrawal, M., Hegselmann, S., Lang, H., Kim, Y. & Sontag, D. Large Language Models are Few-Shot Clinical Information Extractors. (2022) doi:10.48550/ARXIV.2205.12689.

11. Liu, P. *et al.* Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* **55**, 1–35 (2023).

12. Reynolds, L. & McDonell, K. Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm. (2021) doi:10.48550/ARXIV.2102.07350.

13. Mishra, S., Khashabi, D., Baral, C., Choi, Y. & Hajishirzi, H. Reframing Instructional Prompts to GPTk's Language. (2021) doi:10.48550/ARXIV.2109.07830.

14. White, J. *et al.* A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. (2023) doi:10.48550/ARXIV.2302.11382.

15. Liu, V. & Chilton, L. B. Design Guidelines for Prompt Engineering Text-to-Image Generative Models. in *CHI Conference on Human Factors in Computing Systems* 1–23 (ACM, 2022). doi:10.1145/3491102.3501825.

16. Wang, J. *et al.* Prompt Engineering for Healthcare: Methodologies and Applications. (2023) doi:10.48550/ARXIV.2304.14670.

17. Reiss, M. V. Testing the Reliability of ChatGPT for Text Annotation and Classification: A Cautionary Remark. (2023) doi:10.48550/ARXIV.2304.11085.

18. Zuccon, G. & Koopman, B. Dr ChatGPT, tell me what I want to hear: How prompt knowledge impacts health answer correctness. (2023) doi:10.48550/ARXIV.2302.13793.

19. Maus, N., Chao, P., Wong, E. & Gardner, J. Black Box Adversarial Prompting for Foundation Models. (2023) doi:10.48550/ARXIV.2302.04237.

20. Antaki, F., Touma, S., Milad, D., El-Khoury, J. & Duval, R. Evaluating the Performance of ChatGPT in Ophthalmology. *Ophthalmology Science* **3**, 100324 (2023).

21. Ye, J. *et al.* A Comprehensive Capability Analysis of GPT-3 and GPT-3.5 Series Models. (2023) doi:10.48550/ARXIV.2303.10420.

22. OpenAI. API data usage policies. https://openai.com/policies/api-data-usage-policies.

23. OpenAI. Models. https://platform.openai.com/docs/models.

24. How do text-davinci-002 and text-davinci-003 differ? https://help.openai.com/en/articles/6779149-how-do-text-davinci-002-and-text-davinci-003-differ.

25. OpenAI. Model index for researchers. https://platform.openai.com/docs/model-index-for-researchers.

26. Moon, S., Pakhomov, S., Liu, N., Ryan, J. O. & Melton, G. B. A sense inventory for clinical abbreviations and acronyms created using clinical notes and medical dictionary resources. *J Am Med Inform Assoc* **21**, 299–307 (2014).

27. Katz, D. M., Bommarito, M. J., Gao, S. & Arredondo, P. GPT-4 Passes the Bar Exam. *SSRN Journal* (2023) doi:10.2139/ssrn.4389233.

28. Microsoft. Microsoft Azure OpenAI Data Policy. https://learn.microsoft.com/en-us/legal/cognitive-services/openai/data-privacy.

29. Adams, G. LMC shared data. https://github.com/griff4692/LMC/tree/master/shared_data/casi.

Index of Appendices

APPENDIX A – Architectural Details

Models & API

We use 3 models from the OpenAI GPT-3.5 range. OpenAI claim that the models provide superior performance to the earlier GPT-3 models. However, they are not the latest GPT-4 release (to which API access was not available at the time of writing). Each model claims to be an incremental performance on the prior iteration, allowing us to examine any performance improvements between the models. GPT-3.5 models are closed-source, so details of their architecture and training data have not been publicly released.

The models are called via the OpenAI API. API calls are charged per token. The turbo model is $1/10^{th}$ of the price of the davinci models. Temperature setting, which dictates the randomness of the output (from 0 to 1) was set to 0 to maximize reproducibility.

Datasets

We built several datasets from the full CASI set. These are set out in the main paper.

The test involved frequent iterative changes to model and prompt combinations. Because each API call is charged per token, we built a smaller, 'hard' set, containing more challenging instances, so that improvements in prompt performance were visible over a smaller API call. We obtained a set of 200 'hard' instances from Reduced-CASI by using a basic prompt and a mixture of earlier GPT-3 models and current GPT-3.5 models and extracting the failed instances. We build a `small_set` of approximately 10% the size of the `main_set` whilst maintaining longform ratio of instances.

Finally, we build a set containing the single longform 'Fairview Southdale Hospital.' This is a local hospital in Minnesota. Any instances containing a reference to that location were removed. This is a novel, proper noun which we would not expect the model to otherwise predict in the context of medical text with no other identifiers of location. This `rare_set` is used to infer whether the CASI dataset was included in the model data.

We also utilise a custom dictionary developed to map shortforms to longforms and the extracted 'section' headings from the CASI data, both developed by Adams et al.[29]

Problem definition and architecture

Each CASI instance contains a shortform, a target longform, and context. For each shortform, there is finite number of longform options in the dataset. The goal is to correctly expand the shortform into the longform. Similarly to Agrawal, we treat the problem as a multiple choice task.

An example of a CASI entry and the longform options used are shown in the figure below.

sf:	*PDA*
context:	*The patient is status post cardiac arrest in _%#MM#%_, 2003; he had a stent placed in the left circumflex coronary artery and also had a defibrillator placed at that time. OM-1 and **PDA** were occlude, OM-2 was subtotally occluded. His last stress test showed a fixed inferior-wall defect consistent with prior infarct; however, there was a reversible lateral-wall defect; it makes me wonder if his OM-2 lesion is causing ischemia which is subsequently causing his rhythm disturbances.*
options:	*0: 'posterior descending artery',*
	1: 'patent ductus arteriosus',
	2: 'patient-controlled analgesia:PCA'
lf:	*0: 'posterior descending artery'*

An example of a CASI entry and the longform options

Resolver

The longform options are not provided to the model. The output from the models is an unstructured string. To convert the string to one of the multiple choice indices, we adopt the 'resolver' concept from Agrawal.[10] This is a post-processing function which takes the model output string and multiple choice options, and maps the output to an index. Our resolver matches the longest common substring between the model output and a longform map and returns the corresponding index. The index is compared to the corresponding index in the original dataset.

We also adopt and slightly modify a list of canonical forms for each longform in CASI which was manually prepared by Adams.[29] This sf-lf map helps to match model outputs where the answer is ostensibly correct to the human eye, but there is no matching substring with the strict abbreviation expansion. For example, "PM" may return "post-meridiem" (strict expansion match) or "afternoon" (not a strict match).

Batching & Formatting Instruction

The total number of tokens sent to the model (and hence the inference cost, frequency and delay) can be reduced by batching requests together. By batching requests, only a single 'instruction' per batch is required.

The OpenAI APIs do not explicitly support batching. However, with the right combination of instructive prompts and post-processing, several instances can be successfully concatenated into a single request, and successfully parsed.

After some iteration, we settled on the formatting instruction shown below.

"[Instruction] Provide the expanded abbreviations in a numbered list from 1 to [batch size] with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task: [Payload]".

The final instructive prompt we settled on in order to ensure the output from the model could be properly resolved in post-processing.

We note that because the model attends to the entirety of the input prompt, the model's performance may be influenced by this formatting instruction. However, conducting testing on a range of format instructions would

have required separate parser modules to be developed and greatly increased the number of inferences required. It was beyond the scope of this study.

Details of the 'Instruction' and 'Payload' sections were amended in the prompt iterations, described in the main paper and Appendix B.

By way of example, the instruction in the context of a full payload with a batch size of 5, is shown in the figure below, along with the raw model output and the post-processing output:

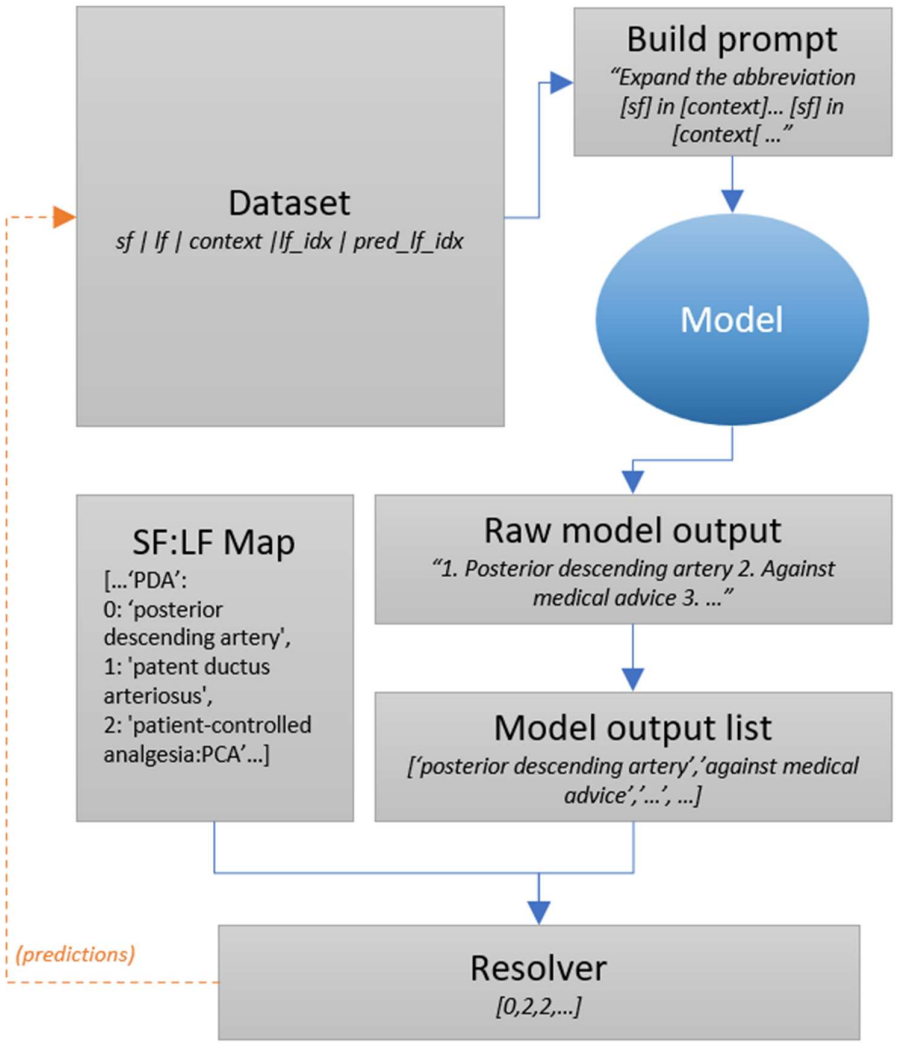
```
PROMPT: Expand the medical abbreviations marked by the characters '~~'
in the following sub-heading and sentence snippets from a doctor's
note. Provide the expanded abbreviations in a numbered list from 1 to 5
with numbers matching the question number marked with (NO_). In your
response state only the expanded abbreviation. Do nothing else! Do not
state the original abbreviation. Do not state your task: (NO_1):
Subheading: 'IMPRESSION/PLAN'. Snippet: '...placed in the left
circumflex coronary artery and also had a defibrillator placed at that
time. OM-1 and  ~~PDA~~  were occlude, OM-2 was subtotally occluded.
His last stress test showed a fixed inferior-wall defect...'.  (NO_2):
Subheading: 'SOCIAL HISTORY'. Snippet: '...industry, however, was laid
off as the contract was lost. He is originally from Kenya. He has been
in the  ~~US~~  at least eight years. He denies any recent travel
history. FAMILY HISTORY: Is noncontributory, specifically...'.  (NO_3):
Subheading: 'HISTORY OF PRESENT ILLNESS'. Snippet: '...discharged in
_%#MM2006#%_ from the University of Minnesota Medical Center, Fairview
with a diagnosis of  ~~SBP.~~  At that time, he presented with
abdominal pain and fevers. He now presents to the emergency room
today...'.  (NO_4): Subheading: 'DISPOSITION/FOLLOWUP'. Snippet:
'...itching. DISPOSITION/FOLLOWUP: The patient left the hospital
against medical advice, refusing to sign the  ~~AMA~~  form. She is to
follow up with her primary care physician, Dr. _%#NAME#%_ _%#NAME#%_.
Please do not...'.  (NO_5): Subheading: 'PHYSICAL EXAMINATION'.
Snippet: '...have limited range of motion. NEUROLOGIC: He is oriented
x1, he is confused secondary to status post  ~~CVA~~  and dementia.
ADVANCED DIRECTIVE: The patient is currently DNR/DNI. PERTINENT
LABORATORY DATA: Sodium...'.

RAW MODEL OUTPUT: 1. posterior descending artery 2. united states 3.
spontaneous bacterial peritonitis 4. against medical advice 5.
cerebrovascular accident

AS LIST: ['posterior descending artery', 'united states', 'spontaneous
bacterial peritonitis', 'against medical advice', 'cerebrovascular
accident']
```

Example of full prompt, raw model response and post-processed output, for batch of size 5.

The final model architecture is shown in the figure below:



The final model architecture used

APPENDIX B – Detailed results of prompt iterations on hard_set

#	Code	Amendment to preceding prompt	Payload	Batch size	Accuracy		
					text-davinci-002	text-davinci-003	gpt-3.5-turbo
1	`prompt = "Expand the abbreviations in the following sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '" + batch.iloc[i]['context'] + "'. "`	N/A	Full, original CASI context	5	0.450	0.635	0.660
2	`prompt = "Expand the abbreviations in the following sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '" + get_reduced_context(batch.iloc[i]['left_tokenized'],batch.iloc[i]['right_tokenized'],batch.iloc[i]['sf_rep'],100,add_marker=False) + "'. "`	Above	Full, preprocessed text (tokenized, removal of punctuation, stop words)	5	0.370	0.515	0.515
3	`prompt = "Expand the abbreviations in the following sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '..." + get_reduced_context_string(batch.iloc[i]['left_string'],batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200,add_marker=False) + "...'. "`	Above	Revert to original CASI text. But reduce window around sf (+- 200 char width either side of sf)	5	0.465	0.670	0.635
4	`prompt = "Expand the medical abbreviations in the following sentence snippets from a doctor's note. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '..." + get_reduced_context_string(batch.iloc[i]['left_string']`	Add medical context: "…Expand the medical abbreviations in the following sentence snippets from a doctor's note…"	Above	5	0.480	0.620	0.590

	,batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200, add_marker=False) + "...'. "					
5	prompt = "Expand the abbreviations marked by the characters '~~' in the following sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:" for i in range(0, len(batch)): prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '..." + get_reduced_context_string(batch.iloc[i]['left_string'] ,batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200, add_marker=True) + "...'. "	Remove medical context. Instruct to expand flagged character in sentence snippet: "Expand the abbreviations marked by the characters '~~' in the following sentence snippets."	Above	5	0.560	0.640	0.710
6	prompt = "Expand the abbreviations marked by the characters '~~' in the following sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:" for i in range(0, len(batch)): prompt += " (NO_" + str(i+1) + "): What does '" + batch.iloc[i]['sf_rep'] + "' in the following sentence refer to?: '..." + get_reduced_context_string(batch.iloc[i]['left_string'] ,batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200, add_marker=True) + "...'. "	Above	Reduced window (+- 100 char width either side of sf)	5	0.545	0.675	0.630
7	prompt = "Expand the abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:" for i in range(0, len(batch)): prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=100,add_marker=True) +"...'. "	Refer to snippet now includes sub-heading: "Expand the abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets."	Sub-heading + Reduced window (+- 100 char width either side of sf)	5	0.600	0.655	0.690
8	prompt = "Expand the abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:" for i in range(0, len(batch)): prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200,add_marker=True) +"...'. "	Above	Reduced window (+- 200 char width either side of sf)	5	0.585	0.630	0.705

9	`prompt = "Expand the abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200,add_marker=True) +"...'. "`	Above	Above	20	0.615	0.680	0.690
10	`prompt = "You are a helpful assistant. Expand the abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200,add_marker=True) +"...'. "`	Add persona: "You are a helpful assistant."	Above	Revert to 5	0.560	0.635	0.695
11	`prompt = "You are a medical professional that expands abbreviations. Expand the medical abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets from a doctor's note. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=200,add_marker=True) +"...'. "`	Add reinforced context of medical abbreviations: "You are a medical professional that expands abbreviations. Expand the medical abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets from a doctor's note."	Above	5	0.590	0.640	0.690
12	`prompt = "You are a medical professional that expands abbreviations. Expand the medical abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets from a doctor's note. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:"` `for i in range(0, len(batch)):` `prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'],`	Above	Change reduced window to 100 char	5	0.590	0.655	0.685

	batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=100,add_marker=True) +"...'. "						
13	Above	Above	Above	20	0.645	0.695	0.710
14	prompt = "~~You are a medical professional that expands abbreviations.~~ Expand the medical abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets from a doctor's note. Provide the expanded abbreviations in a numbered list from 1 to " + str(len(batch)) + " with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:" for i in range(0, len(batch)): prompt += " (NO_" + str(i+1) + "): Subheading: '" + batch.iloc[i]['section'] + "'. Snippet: '..." + get_reduced_context_string(batch.iloc[i]['left_string'], batch.iloc[i]['right_string'],batch.iloc[i]['sf_rep'],width=100,add_marker=True) +"...'. " messages=[{"role": "system", "content": "You are a medical professional who expands abbreviations."}, {"role": "user", "content": prompt}]	Above, but move persona sentence to 'content' for gpt-3.5-turbo.	Above	20	N/A	N/A	0.720

APPENDIX C – Details of the optimal prompt from Experiment 2

Details of the optimal prompt when running on the `small_set`:

Model:	gpt-3.5-turbo
Temperature:	0
Batch Size:	20
Content Message:	"You are a medical professional who expands abbreviations."
Prompt:	“Expand the medical abbreviations marked by the characters '~~' in the following sub-heading and sentence snippets from a doctor's note. Provide the expanded abbreviations in a numbered list from 1 to 20 with numbers matching the question number marked with (NO_). In your response state only the expanded abbreviation. Do nothing else! Do not state the original abbreviation. Do not state your task:”
Payload:	“(NO_X): Subheading: '[subheading]'. Snippet: '...[context +-100 characters around marked ~~SF~~]…’.”

APPENDIX D – Detailed metrics from Experiment 3

Detailed metrics for each shortform run using the optimum prompt on the `main_set` are shown the in table below.

(where w = weighted, m = macro, totals are sum or mean).

shortform	count	targets	acc	mPr	mR	mF1	wPr	wR	wF1
AMA	471	3	0.9406	0.8066	0.8106	0.8054	0.9394	0.9406	0.9394
ASA	395	2	0.9975	0.875	0.9987	0.9279	0.9981	0.9975	0.9976
AV	491	3	0.9674	0.8637	0.9268	0.8886	0.9696	0.9674	0.968
BAL	485	2	0.9423	0.8426	0.7689	0.8001	0.9375	0.9423	0.9388
BM	488	3	0.959	0.785	0.8566	0.8151	0.9626	0.959	0.9603
C&S	432	5	0.625	0.3773	0.4389	0.3429	0.9157	0.625	0.7062
CEA	497	4	0.9598	0.4677	0.4649	0.4663	0.9697	0.9598	0.9647
CR	499	6	0.8818	0.381	0.7207	0.4173	0.9519	0.8818	0.9081
CTA	495	4	0.9717	0.473	0.4867	0.4795	0.9713	0.9717	0.9712
CVA	474	2	0.9641	0.9656	0.9614	0.9633	0.9644	0.9641	0.9641
CVP	487	3	0.9569	0.6141	0.6242	0.619	0.9616	0.9569	0.9592
CVS	237	3	0.9536	0.9295	0.8162	0.8406	0.9634	0.9536	0.9551
DC	455	5	0.7626	0.5405	0.828	0.5636	0.8886	0.7626	0.7924
DIP	492	3	0.9776	0.9252	0.9834	0.9507	0.9817	0.9776	0.9788
DM	484	3	0.938	0.6271	0.6276	0.6272	0.9429	0.938	0.9401
DT	475	6	0.8589	0.4743	0.611	0.4983	0.9198	0.8589	0.8823
EC	473	4	0.9429	0.6203	0.9146	0.6507	0.9601	0.9429	0.9489
ER	495	3	0.9657	0.8232	0.9691	0.8773	0.9771	0.9657	0.9691
FSH	265	2	0.9962	0.9981	0.8333	0.899	0.9962	0.9962	0.9959
IA	171	2	0.9591	0.6111	0.9793	0.6712	0.9909	0.9591	0.9717
IM	492	2	0.9878	0.9465	0.9686	0.9572	0.9882	0.9878	0.988
LA	454	3	0.663	0.4062	0.8583	0.3901	0.9413	0.663	0.7488
LE	481	7	0.9002	0.406	0.5214	0.4294	0.9383	0.9002	0.9144
MR	492	5	0.9512	0.5731	0.849	0.6152	0.9744	0.9512	0.9616
MS	488	6	0.9549	0.5095	0.8191	0.5591	0.9783	0.9549	0.965
NAD	465	2	0.7505	0.6389	0.665	0.648	0.7763	0.7505	0.7612
NP	463	4	0.9546	0.7621	0.9379	0.7742	0.9623	0.9546	0.9572
OP	489	6	0.8405	0.4932	0.7294	0.5425	0.8839	0.8405	0.8531
PA	412	6	0.9296	0.7885	0.8075	0.7962	0.9404	0.9296	0.9325
PCP	488	4	0.9385	0.7811	0.9542	0.8243	0.9455	0.9385	0.9401
PDA	478	3	0.8159	0.5787	0.9069	0.6126	0.8825	0.8159	0.8279
PM	375	3	0.7653	0.4415	0.4512	0.4451	0.7895	0.7653	0.7762
PR	241	4	0.9502	0.9288	0.9215	0.9238	0.9537	0.9502	0.9503
PT	496	4	0.9153	0.5467	0.6484	0.5619	0.9602	0.9153	0.9302
RA	490	4	0.9571	0.6822	0.7245	0.7008	0.965	0.9571	0.9594
RT	470	4	0.8532	0.6792	0.7511	0.581	0.9523	0.8532	0.8852
SA	454	5	0.8106	0.4247	0.8555	0.416	0.9711	0.8106	0.8788

SBP	489	2	0.9591	0.9076	0.9608	0.9313	0.964	0.9591	0.9604
US	290	2	0.9759	0.9795	0.9652	0.9719	0.9762	0.9759	0.9757
VAD	482	4	0.888	0.5273	0.8179	0.5656	0.9308	0.888	0.9051
VBG	483	2	0.8406	0.8478	0.858	0.84	0.868	0.8406	0.8417
total/mean:	**18233**	**150**	**0.9054**	**0.6793**	**0.7949**	**0.6876**	**0.944**	**0.9054**	**0.9177**

APPENDIX E – Experiment 4 string counts

Counts of predicted references to different strings for the acronym 'FSH' are shown in the table below. The true shortform is 'fairview southdale hospital'.

String predicted by model	actual	predicted	
		batch size: 5	batch size: 20
fairview southdale hospital	***190***	***71***	***109***
florida surgical hospital	0	15	20
florida state hospital	0	10	0
fort sanders hospital	0	16	19
fort sam houston	0	14	20
fred hutchinson cancer research center	0	0	14
other	0	64	8
Total:	**190**	**190**	**190**

APPENDIX F – Source Code

Please refer to the attached: “AbbreviationExpansionCode.zip” and the included readme.MD.