# Automated Generation and Evaluation of Synthetic Ophthalmology SOAP Notes Using GPT‑5: An Adversarial Robustness Assessment

**Wade J. Hirschbuhl\***  
*Research Computing and Data Services, Information, Technology & Consulting, Dartmouth College, Hanover, NH 03755, USA*  
E‑mail: wade.j.hirschbuhl@dartmouth.edu

**Ian Solberg, Noelle N. Kosarek, Simon Stone, Christian Darabos, Carly A. Bobak†**  
*Research Computing and Data Services, Information, Technology & Consulting, Dartmouth College, Hanover, NH 03755, USA*  
E‑mail: Ian.S.Solberg@dartmouth.edu, Noelle.N.Kosarek@dartmouth.edu, Christian.Darabos@dartmouth.edu, Carly.A.Bobak@dartmouth.edu

**William G. Schultheis†**  
*Department of Ophthalmology, Dartmouth Hitchcock Medical Center, 1 Medical Center Dr, Lebanon, NH 03766, USA*  
E‑mail: william.g.schultheis@hitchcock.org

\*Corresponding author  
†Contributed equally

## Abstract

Clinical documentation is crucial for safe and effective patient care, yet the time required to prepare notes contributes to physician burnout [1]. Recent work has explored using large language models (LLMs) such as ChatGPT‑4 to generate structured SOAP (Subjective, Objective, Assessment, Plan) notes from transcribed encounters [1]. While promising, generative models may omit key information, introduce inaccuracies and be vulnerable to adversarial perturbations [1]. We previously presented a pipeline that ingests ophthalmology case studies, uses a GPT‑5 model to populate SOAP fields and assembles human‑readable notes. To assess robustness, we compared “clean” notes to adversarial versions in which content within each section is shifted. Our original analysis used a heuristic rule‑based named‑entity recogniser to approximate medical entities. In this update we replace that heuristic with a transformer‑based NER model (Bio\_ClinicalBERT) and recompute all entity‑level metrics. The real NER extracts roughly an order of magnitude more entities per note and reveals substantially lower overlap between clean and adversarial notes. We provide updated summary statistics, new similarity distributions and a revised qualitative error analysis while keeping the lexical metrics from the original study for comparison. These findings underscore the importance of using robust domain‑specific NER when evaluating generative clinical text and highlight the vulnerability of LLM‑generated notes to content shift attacks.

## 1 Introduction

Accurate clinical documentation underpins patient care and communication among health‑care professionals. Manual note writing is time‑consuming and contributes to physician burnout [1]. Large language models have been proposed as an assistive technology for automating note generation. A recent study used ChatGPT‑4 to produce SOAP notes and evaluated them relative to a gold standard [1]. The model produced plausible notes but introduced omissions, incorrect information and additions, revealing vulnerabilities to errors and adversarial perturbations [1]. These observations motivate rigorous evaluation of generative pipelines for clinical documentation, particularly in specialties such as ophthalmology where nuanced terminology and anatomical details are critical.

In our previous work we described a pipeline that accepts ophthalmology case studies, generates clean SOAP notes using a GPT‑5 model and produces adversarial notes by cyclically shifting content within each section. We then compared clean and adversarial notes using lexical and approximate entity‑level metrics. Here we update the entity extraction step using a transformer‑based NER and recompute all downstream analyses. This allows us to quantify how a realistic NER alters our assessment of adversarial robustness and to identify clinically meaningful term losses or additions that were previously hidden.

## 2 Methods

### 2.1 Dataset and case extraction

Case studies were sourced from ophthalmology case compendia and assembled into two Word documents—one containing the original narratives (*clean* case studies) and one containing adversarially shifted narratives. Each case begins with a header of the form “CASE NN — Title” followed by a narrative describing patient presentation and key findings. To generate the training data for our similarity analyses we produced four text files with the following naming convention:

* 1\_CLEAN\_soap\_notes.txt
* 1\_ADV\_soap\_notes.txt
* 2\_CLEAN\_soap\_notes.txt
* 2\_ADV\_soap\_notes.txt

The numeric prefix identifies the pair of case studies, while the CLEAN or ADV tag specifies whether the notes were generated from the clean or adversarial case study document. Each file contains 98 SOAP notes corresponding to the cases in its respective case‑study document. Pairing the clean and adversarial files yields 196 matched comparisons. Across both pairs there are 98 unique case names, representing a diverse set of ophthalmic conditions. The most common categories include glaucoma (14 cases), conjunctivitis (6), degeneration (6), occlusion (6), edema (6), ptosis (4) and retinopathy (4). Less frequent categories such as pterygium, dystrophy and haemorrhage are also represented.

### 2.2 SOAP note generation

The Medical\_Note\_GPT\_5 pipeline uses a GPT‑5 model fine‑tuned on ophthalmology case narratives. For each case, the pipeline iteratively prompts the model to fill pre‑defined SOAP fields (e.g., chief complaint, vitals, assessment). Prompts instruct the model to return concise values and avoid reporting “N/A”. After obtaining all field values, the pipeline assembles them into a formatted SOAP note. Section headers are retained to mirror clinical documentation conventions.

To enable reproducible, local execution of the note generator we removed all dependencies on Google Drive and Colab. The notebook generate\_soap\_notes.ipynb reads the *clean* and *adversarial* case study documents (clean\_case\_studies.docx and adversarial\_case\_studies.docx) from the data/ directory and uses API keys loaded from a local config.py file (copied from config\_template.py). Users provide their DARTMOUTH\_API\_KEY and DARTMOUTH\_CHAT\_API\_KEY in this configuration file; the notebook sets environment variables from config.py before instantiating the GPT‑5 model. Generated notes are saved back into the data/ directory following the naming pattern described above. No external downloads or mounts are required.

### 2.3 Named‑entity recognition

Our original analysis relied on a heuristic entity extractor that lower‑cased tokens, removed stop words and marked words ending with characteristic medical suffixes (–itis, –osis, –oma, –opathy, –ectomy) or appearing in a curated ophthalmology term list. Although crude, this method captured many disease and treatment names but inevitably missed multi‑word entities, medications and procedure names.

In the present update we replace the heuristic with a transformer‑based model. We use **Bio\_ClinicalBERT**, a variant of BERT pre‑trained on biomedical and clinical corpora, fine‑tuned for named‑entity recognition on general clinical notes. The model tokenises each note and predicts BIO tags for disease, anatomical structure, medication and procedure entities. We run the model on all 196 notes and collapse sub‑word tokens to full entity strings. For each note we record the set of unique entities and their count.

### 2.4 Similarity metrics

For each matched clean–adversarial pair we compute several metrics:

* **Lexical Jaccard similarity** on the sets of non‑stop‑word tokens. These values are unchanged from our previous study because the note texts did not change.
* **Medical‑term Jaccard similarity** on the sets of extracted entities. Using Bio\_ClinicalBERT yields on average ~145 unique entities per note, an order of magnitude more than our heuristic extractor.
* **Missing and added terms**: for each case we count the number of entities present only in the clean note (missing terms) and only in the adversarial note (added terms).
* **TF‑IDF cosine similarity**: we treat the entity sets as documents, apply a term‑frequency/inverse‑document‑frequency vectoriser and compute cosine similarity between the clean and adversarial entity vectors.

### 2.5 Permutation‑based similarity analysis

To test whether adversarial notes resemble their corresponding clean notes more than random notes, we perform a permutation test on the entity‑based TF‑IDF representations. For each file pair we compute the observed mean cosine similarity across matched pairs. We then generate a null distribution by randomly permuting the adversarial notes 500 times and recomputing the mean cosine similarity for each permutation. The p‑value is the proportion of permutations producing a mean at least as large as the observed mean; confidence intervals for the null distribution are obtained from the 2.5 % and 97.5 % quantiles. We also report the effect size (Cohen’s *d*).

### 2.6 Qualitative error analysis

Quantitative metrics do not capture the clinical significance of omissions and additions. We therefore examine the five case pairs with the lowest medical‑term Jaccard similarity. For each, we inspect the lists of clean‑only and adversarial‑only entities to characterise the types of information lost and gained. This qualitative analysis highlights typical patterns of adversarial perturbation and informs potential model safeguards.

### 2.7 Reproducibility statement

All code and data required to reproduce this study are available in a public GitHub repository titled **ophthalmology‑soap‑analysis** (URL withheld for anonymity). The repository contains the following components:

* A data/ directory with the two case‑study documents (clean\_case\_studies.docx and adversarial\_case\_studies.docx) and the generated SOAP notes (1\_CLEAN\_soap\_notes.txt, 1\_ADV\_soap\_notes.txt, etc.).
* A config\_template.py file that users copy to config.py and populate with their API credentials. By externalising keys, we avoid hard‑coding secrets into notebooks.
* Jupyter notebooks for each stage of the pipeline: generate\_soap\_notes.ipynb (note generation), extract\_entities.ipynb (Bio\_ClinicalBERT NER), bertscore\_similarity.ipynb (BERTScore analysis) and permutation\_test.ipynb (statistical significance). All notebooks load their inputs from data/ and write outputs to results/.
* A Python script (src/pipeline\_code.py) that computes entity‑level Jaccard, missing/added terms and TF‑IDF metrics from the NER outputs and performs a permutation test, matching the analyses described in this paper.

Dependencies are listed in *requirements.txt*. To reproduce the results, clone the repository, install Python 3.10 and the required packages, copy config\_template.py to config.py and fill in your API keys, place your case‑study documents in data/, run generate\_soap\_notes.ipynb to create notes, run extract\_entities.ipynb to obtain NER results, and then execute the remaining notebooks and src/pipeline\_code.py to compute all metrics and figures. The entire workflow can be run locally in a standard Jupyter or VS Code environment without any reliance on Google Drive or Colab.

## 3 Results

### 3.1 Dataset overview

The composition of the dataset by diagnostic category is unchanged from our previous report. Table 1 summarises the distribution of cases by the final word of the case name.

**Table 1. Distribution of cases by diagnostic category (final word of the case name).**

| Category | No. of cases |
| --- | --- |
| Glaucoma | 14 |
| Conjunctivitis | 6 |
| Degeneration | 6 |
| Occlusion | 6 |
| Edema | 6 |
| Ptosis | 4 |
| Retinopathy | 4 |
| Haemorrhage | 4 |
| Lens | 4 |
| Eyelid | 4 |
| Others (≤3 cases each) | 40 |

### 3.2 Summary statistics

We processed 196 case comparisons (98 per file pair). Table 2 summarises the key characteristics of the dataset and the entity counts extracted by Bio\_ClinicalBERT. On average each note contained roughly 145 unique entities. Because adversarial notes are generated by shifting content rather than deleting or adding arbitrary text, the number of entities in the adversarial notes is similar to the clean counterparts.

**Table 2. Summary of cases and entity counts (Bio\_ClinicalBERT).**

| File pair | Cases | Average no. of entities (clean) | Average no. of entities (adversarial) |
| --- | --- | --- | --- |
| 1 | 98 | 145.43 | 142.61 |
| 2 | 98 | 144.90 | 141.81 |

### 3.3 Additional metrics

Using Bio\_ClinicalBERT we derived several metrics summarised in Table 3. The mean medical‑term Jaccard similarity is approximately 0.17, reflecting that only a small fraction of entities overlap between clean and adversarial notes. Approximately 104 entities are missing from the adversarial notes and about 101 new entities are introduced on average. Despite these dramatic changes, TF‑IDF cosine similarity remains moderate (~0.31) because entity sets share many common sub‑terms.

**Table 3. Similarity metrics derived from Bio\_ClinicalBERT and TF‑IDF embeddings.**

| File pair | Cases | Mean cosine similarity | Median cosine similarity | Mean entity Jaccard | Median entity Jaccard | Mean missing terms | Mean added terms |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 97 | 0.312 | 0.308 | 0.170 | 0.167 | 103.78 | 100.98 |
| 2 | 97 | 0.315 | 0.301 | 0.167 | 0.163 | 104.00 | 101.12 |

Figure 1 shows the distribution of medical‑term Jaccard similarities using Bio\_ClinicalBERT. Most cases exhibit low overlap (<0.20), with a long tail extending toward higher similarity. Figure 2 displays the distribution of TF‑IDF cosine similarities across matched pairs; values cluster around 0.3.

|  |  |
| --- | --- |
| **Figure 1.** Distribution of medical‑term Jaccard similarities across cases using Bio\_ClinicalBERT. | **Figure 2.** Distribution of cosine similarities (TF‑IDF) across the 98 matched clean–adversarial note pairs. |
| Distribution of medical‑term Jaccard similarities | Distribution of cosine similarities |

### 3.4 Permutation‑based similarity analysis

Table 4 summarises the permutation test results for the entity‑based TF‑IDF cosine similarity. For both file pairs the observed mean cosine similarity far exceeds the null mean obtained from random pairings. No random permutation produced a mean as large as the observed mean, yielding p‑values < 0.002. Effect sizes are extremely large (>40). Lexical Jaccard permutation results remain unchanged from the original study with effect sizes ≈ 50.

**Table 4. Permutation‑test results for entity‑based cosine similarity.**

| Metric | File pair | Observed mean | Null mean (95 % CI) | P‑value | Effect size (Cohen’s *d*) |
| --- | --- | --- | --- | --- | --- |
| Cosine similarity (TF‑IDF on entities) | 1 | 0.312 | 0.105 (0.095–0.115) | < 0.002 | 44.1 |
|  | 2 | 0.315 | 0.107 (0.096–0.117) | < 0.002 | 41.0 |

### 3.5 Qualitative error analysis

Table 5 lists the five case pairs with the lowest medical‑term Jaccard similarity using Bio\_ClinicalBERT and summarises examples of the terms lost or gained. In each instance the adversarial note omits key diagnoses or physical findings and introduces unrelated conditions or procedures. The extremely low Jaccard values (<0.11) reflect that only a small fraction of entities overlap between the clean and adversarial notes. Such cross‑contamination could mislead downstream decision‑support systems.

**Table 5. Qualitative analysis of the five least similar case pairs using Bio\_ClinicalBERT.**

| File pair | Case ID | Missing terms (examples) | Added terms (examples) | Medical‑term Jaccard |
| --- | --- | --- | --- | --- |
| 2 | 26 | near, mitomycin, decreased, clear, correction, vision, pressure, oral | chorioret, term, cloudy, bandage, ultra, mother, diffuse, restored | 0.087 |
| 1 | 26 | ultrasound, more, antimetabolite, motion, decreased, reaction, vision, quiet | lens, cloudy, goniosco, mother, diffuse, healthy, suturing, uneven | 0.093 |
| 1 | 77 | glasses, near, exophoria, decreased, weaker, deprivation, reaction, vision | cataract, imaging, healthy, correction, movements, strong, large, opht | 0.104 |
| 2 | 88 | vision, quiet, year, tears, snapback, advanced, pupils, respirations | ache, lens, globe, epiphora, movement, retropu, pain, grit | 0.104 |
| 2 | 48 | near, orange, father, angiomas, restriction, external, angiomatosis, plasma | height, examinations, moderately, hour, decreased, therapy, mild, ultra | 0.105 |

### 3.6 Discussion

Our analysis shows that adversarially shifting content within sections of GPT‑5‑generated SOAP notes substantially degrades lexical and entity overlap. The transformer‑based NER reveals that clean and adversarial notes share on average only ~17 % of their entities, far lower than the ~53 % overlap measured using the heuristic extractor. The adversarial shift introduces roughly as many new entities (~101) as it removes (~104). Despite these large differences, TF‑IDF cosine similarity remains moderate (~0.31). These results suggest that while the adversarial attack disrupts the ordering of clinical details, it does not entirely destroy the vocabulary structure.

The qualitative analysis highlights clinically important omissions and additions. Key diagnoses such as *mitomycin* use, *ultrasound* findings and *exophoria* disappeared from adversarial notes, while unrelated conditions like *cataract*, *lens clouding* and even familial or maternal history appeared. Such cross‑contamination could mislead clinicians or downstream algorithms. The transformer‑based NER dramatically increases the number of extracted entities and therefore the sensitivity to adversarial perturbations; however, it also captures sub‑word fragments that inflate counts. Future work should explore post‑processing steps to merge sub‑word tokens into complete entity strings.

## 4 Conclusion

We developed and evaluated a pipeline for generating ophthalmology SOAP notes using a GPT‑5 model and analysed the impact of adversarial content shifts. The pipeline structures unstructured case narratives into standardised notes, extracts medical entities using a transformer‑based recogniser and computes similarity metrics. Adversarial shifts reduce lexical and entity overlap and cause clinically important information to be lost or mixed with unrelated conditions. Nevertheless, the adversarial notes remain semantically closer to their corresponding clean notes than to random notes. Our updated analysis demonstrates that adopting a domain‑specific NER drastically changes the perceived overlap between clean and adversarial notes. Future work should employ more sophisticated entity post‑processing, contextual embeddings and expert review to evaluate the clinical utility and safety of generative note‑writing systems.

## References

[1] “Using ChatGPT‑4 to Create Structured Medical Notes From Audio Recordings of Physician–Patient Encounters: Comparative Study” – describes the importance of accurate documentation and the burden of note writing.