

Clinical NLP Fundamentals

Study Guide

by Richard J. Young, Ph.D.

richard@deepneuro.ai

Chapter 17 of "Healthcare Analytics and AI: Building Systems That Actually Work"

The essentials: what to know, why it matters at the bedside, and how models fail.

One-sentence thesis. A clinical NLP pipeline is a reduction engine that turns narrative into computable codes, and the practitioner's real skill is knowing exactly what the pipeline throws away.

PART 1: Why Clinical Text Breaks Standard NLP

The 80 Percent Problem

Roughly **80 percent** of EHR data lives as unstructured free text: physician notes, nursing assessments, radiology and pathology reports, discharge summaries. Structured fields record what was decided; the narrative records how and why, and often facts the codes omit entirely. A model trained only on structured data therefore learns billing reality, not clinical reality, because social context, diagnostic uncertainty, and the physician's differential all hide in the prose.

ANALOGY

The structured chart is the receipt. The note is the conversation that produced it. You can reconstruct a lot from receipts, but not the reasoning, the hesitation, or the patient's own words.

The Four Traps of Clinical Language

Clinical text violates nearly every assumption a general NLP pipeline makes. **Abbreviations** are the first

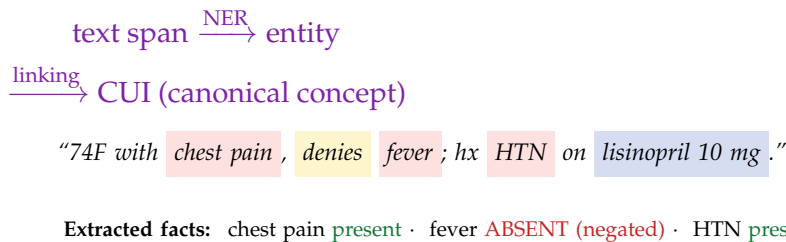
trap: over 170,000 distinct clinical abbreviations exist, and one string carries many meanings, so "MS" can be multiple sclerosis, mitral stenosis, morphine sulfate, mental status, or musculoskeletal. **Negation** is the second: physicians document pertinent negatives, and in one MIMIC-III study over **40 percent** of "pneumonia" mentions were negated ("no evidence of pneumonia"). **Uncertainty and hedging** is the third: "likely pneumonia," "cannot rule out PE," and "questionable infiltrate" all sit between present and absent. **Context-dependent meaning** is the fourth: "father had colon cancer" is family history, "might need a colectomy" is hypothetical, and "treated at an outside hospital" happened elsewhere. A pipeline that extracts "chest pain" without catching "denies chest pain" has reversed the clinical meaning of the note, in the exact direction that triggers unnecessary treatment and inflates risk scores.

Watch out. A naive keyword search inflates prevalence, because the negation, not the noun, carries the clinical truth.

NER and the UMLS

Once you accept that the note carries the clinical truth, the first engineering question is blunt: how do you turn “MI” scrawled in a hurried sentence into something a computer can count across a million charts? It takes two steps. Named entity recognition finds and classifies the spans that are medications, diseases, symptoms, procedures, labs, and anatomy; entity linking then maps each span to a canonical concept.

In symbols, the pipeline reduces free text to a stable identifier in two hops:



NER tags the entity spans (problem , medication , negation trigger); assertion classification then sets their status. “denies fever” is detected as an entity but resolves to absent, so a keyword counter that ignores negation records the exact opposite of the note’s meaning.

Rule-Based vs Machine Learning NER

Two lineages solve NER differently. **Rule-based** systems (MetaMap, cTAKES) do dictionary lookup against the UMLS with heuristic rules: comprehensive and interpretable, but slow, format-sensitive, and weak on ambiguous abbreviations, where one benchmark reported $F = 0.165$. **Machine learning** moved from conditional random fields (CRFs) to transformers: ClinicalBERT (pretrained on MIMIC notes), GatorTron (8.9 billion parameters, 90 billion words), and Clinical ModernBERT (8,192-token context, so a full discharge summary is not truncated). Rule-based systems are strong on what has already been named and cataloged; ML systems handle the messy shorthand no dictionary anticipated. But bigger is not automatically better: zero-shot GPT-4 scores substantially lower than a fine-tuned ClinicalBERT on token-level NER, because precise character-level entity boundaries are not what generative pattern completion is good at.

Encoder vs Generative Extraction, and Embeddings

Two jobs need two different tools. For extraction, the task is to pull exact character-level spans, and fine-

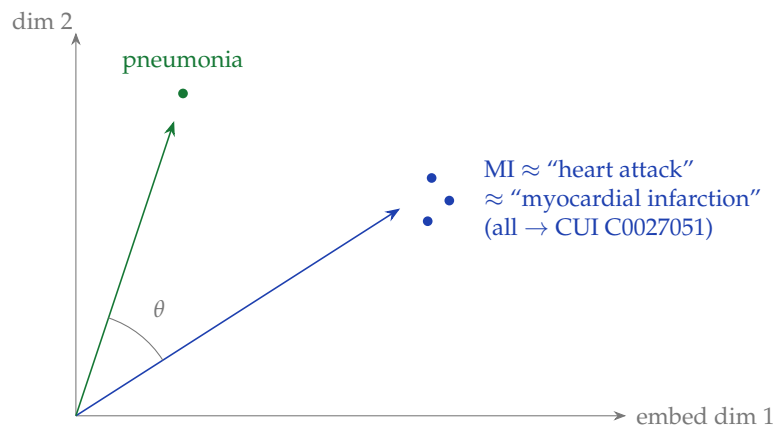
The Unified Medical Language System (UMLS) is the metathesaurus that makes this possible: over 200 source vocabularies, 4.4 million concepts, and a Concept Unique Identifier (CUI) per concept, so “heart attack,” “MI,” and “myocardial infarction” all normalize to CUI C0027051. Normalization is what lets you count the same condition across notes, patients, and hospitals; without it every synonym is a separate, uncountable event, and the UMLS is the master card catalog that points thousands of local labels, abbreviations, and synonyms back to one indexed concept.

tuned **encoder** models (the BERT family) still beat **generative** LLMs at it. Retrieval is the opposite problem: you want the passage that answers a query even when it shares no words with it, so the real question is how to measure closeness in meaning rather than in spelling. The trick is to turn text into embeddings and rank passages by the angle between their vectors, cosine similarity:

$$\text{sim}(q, d) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\| \|\mathbf{d}\|}$$

Depth beats breadth: CardioEmbed, trained on cardiology textbooks, hit 99.60 percent retrieval accuracy on cardiology tasks, a +15.94 point gain over the broad-corpus MedTE (83.66 percent). A model trained deeply on one specialty is a hospitalist; a broad medical embedding is a generalist who knows a little of everything and the precise thing poorly. For a cardiology RAG system a general embedding retrieves plausible but wrong passages, so the right architecture is an ensemble of deep specialists, not one shallow generalist. That specialization has a cost: CardioEmbed’s general-medical retrieval dropped

sharply (0.20 NDCG@10), because a single model cannot maximize depth and breadth at once.



Embedding space: synonyms for one concept collapse to nearly the same vector (small angle θ , cosine similarity near 1) and normalize to a single CUI, while an unrelated concept points elsewhere. Retrieval ranks passages by this angle.

Meaning Versus Billing

SNOMED CT captures clinical *meaning*; ICD-10-CM is required for *billing*. The rule to memorize: encode in SNOMED, then map to ICD-10 to preserve granularity while meeting regulation. These systems overlap, conflict, and drift, so a pipeline that maps to the wrong system, or uses a stale mapping, produces data

downstream systems cannot interpret; versioning and mapping currency are operational requirements, not footnotes. ICD-10 codes are long-tailed, though: a few codes (diabetes, hypertension, heart failure, COPD) dominate while tens of thousands appear rarely, a class-imbalance problem that keeps automated coding an open research area despite a 46 percent median coding-time reduction in a 2024 trial.

Coding systems at a glance

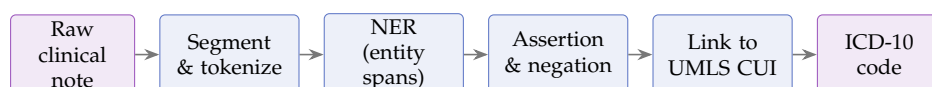
System	Maintained by	Size	Primary use	NLP challenge
ICD-10-CM	NCHS / CMS	~72,000 codes	Billing diagnoses	Long-tail imbalance
CPT	AMA	~10,000 codes	Procedure reimbursement	Op-note extraction
SNOMED CT	SNOMED Intl	350,000+ concepts	Clinical meaning in EHRs	Rich ontological linking
LOINC	Regenstrief	100,000+ codes	Lab interoperability	Value normalization
RxNorm	NLM	~300,000 strings	Medication interoperability	Drug name normalization

A Sequence, Not a Model

A production pipeline is an orchestrated sequence, not a single model, and failures at any stage propagate downstream. The reason the whole apparatus exists comes down to one equation: a concept the NER stage detected is not yet a clinical fact until you also know whether it is being affirmed, denied, or merely entertained.

concept + assertion status = clinical fact

Assertion status (present, absent, possible, historical, hypothetical, family) is what turns a detected concept into a usable fact: the pipeline is a refinery that takes something clinically valuable but messy, crude narrative, and turns it into forms downstream systems can burn.



The clinical-NLP pipeline is an orchestrated sequence, not one model: narrative enters at the left and leaves as a billable code. A failure at any stage propagates downstream, and each stage discards detail the codes cannot hold.

Negation and Assertion: NegEx and ConText

NegEx (Chapman et al., 2001) flags any concept inside a fixed token window after a trigger (“no,” “denies,” “ruled out”): fast, interpretable, and effective, at 94.5 percent specificity on discharge summaries. **ConText** (Harkema et al., 2009) added three dimensions, experienter (patient vs family), temporality (current, historical, hypothetical), and certainty, and those four

dimensions are the difference between “the patient has pneumonia” and “the patient’s father had pneumonia and ours is ruled out”, same tokens, opposite meaning. Fixed windows do ignore syntax, though: “tested for HIV but the result was not available” needs “not” to negate “available,” not “HIV.” Fine-tuned assertion models now reach 96.2 percent, but rule-based systems keep the edge on interpretability and deployability where explainability is required.

Pipeline stages at a glance

Stage	Function	Representative tool	Example output
Preprocess / segment	Sentence + token boundaries	medspaCy	Sentence list
Section detection	Tag HPI, A&P, Meds	medspaCy sectionizer	Section-labeled spans
NER	Identify entities	cTAKES, ClinicalBERT	Entity spans + types
Negation / assertion	Status classification	NegEx, ConText	present / absent / possible
Entity linking	Map to UMLS / SNOMED	MedCAT	CUI + canonical name
Relation extraction	Concept-to-concept	Fine-tuned transformer	(drug, dose, condition)

What the Pipeline Misses

The note your pipeline consumes was written by an exhausted human. Physicians spend only **27.2 of 57.8** weekly hours on direct patient care, and 16 percent cite documentation as the single biggest driver of burnout, so under pressure a rich narrative (“74-year-old retired schoolteacher, lives alone, recently widowed, frightened”) compresses to “74F w/ DOE x3d. Lives alone.” The facts survive; the story dies. Social determinants (insurance loss, cost-driven non-adherence), temporal reasoning (a five-year gap between cardiac events), causal chains, and emotional state all live in narrative and fall through every ex-

traction category, so a readmission model built on extracted structured data treats an incomplete record as complete truth.

ANALOGY

The pipeline photographs the skeleton perfectly and misses that the patient is terrified, broke, and alone.

Watch out. Ambient AI (Dragon Copilot, formerly DAX, now over 100,000 clinicians across 600+ health systems) is generating the notes, so future pipelines will extract structured data from text a language

model wrote, a recursive loop that can hallucinate findings and smooth over the very uncertainty clinicians documented on purpose.

| Should You Build This?

Build it when the facts you need are locked in the free text that structured fields never captured, roughly 80 percent of the EHR, and a human still reviews what the pipeline extracts: the 46 percent cut in median coding time was measured with coders in the loop, not removed from it. **Think twice when** pipeline

output feeds downstream decisions without negation and assertion handling, or when the signal you need is the narrative context, social isolation, functional decline, that a reduction engine strips out by design.

Who bears the risk. Patients first, because extraction errors run in the direction of treating documented absence as presence, inflating risk scores and triggering unnecessary care; physicians carry the rest, since the notes your model consumes were written under a documentation burden the pipeline inherits but cannot see.

Quick Self-Test

1. A keyword search returns 36,000 notes mentioning “pneumonia.” Why is that number almost certainly wrong, and what algorithm fixes it?
2. What does a CUI give you that a raw text span does not?
3. Why does a fine-tuned ClinicalBERT beat zero-shot GPT-4 on token-level NER?
4. Name the three dimensions ConText adds on top of NegEx’s negation, and why each one changes clinical meaning.
5. In one sentence, why is CardioEmbed’s poor general-medical retrieval a feature, not a bug?
6. Give two clinically significant things a strong NER pipeline still cannot extract from a discharge summary.

If You Remember One Thing

Extraction is subtraction. The pipeline preserves what can be categorized (entities, codes, assertion status) and loses what often determines whether care succeeds: how the patient got here, the barriers they face, and the reasoning behind the plan. The best practitioners are not the ones with the highest F-score; they are the ones who design systems that surface what was thrown away instead of hiding it.

Richard J. Young, Ph.D. · richard@deepneuro.ai · Senior AI Research Scientist, UnitedHealth Group · UNLV Lee Business School · ORCID 0000-0002-1109-7552