Using Large Language Models for Call Center Transcript Analysis

Leveraging LLMs for call center transcripts enables automated conversational intelligence: identifying outcomes, customer sentiment, script compliance, and actionable insights from conversations [1] [2]. By processing transcripts with well-designed prompts, businesses can spot trends (common complaints, churn risk), evaluate agent performance, and summarize long dialogues into key points [1] [2]. Below is a comprehensive guide to building such LLM-based analytics, including preprocessing, advanced prompting strategies, and quality checks.

Preprocessing Transcripts

Effective analysis begins by preparing the transcript data:

- **Speaker Segmentation & Labeling:** Separate the transcript by speaker turns and label them explicitly (e.g. prepend each line with “Agent:” or “Customer:”). Explicit diarization helps the model understand context [3] [4]. For example, Snowflake’s prompt specifies “A denotes the Agent and C denotes the Customer” to clarify roles [4]. Labeling speakers has been shown to improve accuracy (up to ~9% in simple tasks [3]).

- **Chunking Long Transcripts:** Long calls (e.g. 30–60+ minutes) often exceed an LLM’s context window. Break the text into manageable segments (by time or by conversational break) and analyze sequentially. As one case noted, a 40-minute call was ~100 pages – too large to input at once [5]. Summarize or extract information on each chunk, then combine results. This reduces context overload and cost [5].

- **Noise Removal:** Clean the text by removing filler words (“um”, “uh”), stutters, or obvious transcription errors. Glyph AI recommends “scrubbing” such noise so the LLM works with focused data [6]. This reduces confusion and improves precision.

- **Optional Retrieval Preparation:** If you plan to use retrieval (see below), index any relevant knowledge (e.g. company policy documents, product FAQ) into a vector store so that relevant facts can be fetched during analysis.

Handling Script Formats: Structured vs. Free-Form

Call center transcripts may follow a **structured script** (a defined agent dialog) or be free-flowing:

- **Structured Script:** If agents follow a prescribed script, encode those requirements in your analysis. For instance, provide the script outline or key mandatory phrases in the prompt or via retrieval. You can ask the LLM to compare the transcript against this script. Example prompt (with placeholders `{SCRIPT}` and `{TRANSCRIPT}`):

```
Prompt: “You are a compliance auditor. Given the required agent script below and
the call transcript, check whether the agent followed each step. For each key
step in the script (e.g. greeting, disclosure, problem resolution, closing),
output `true` or `false`, and explain any omissions.
Script: {SCRIPT}
Transcript: {TRANSCRIPT}”
```

This leverages the known structure. Techniques like Retrieval-Augmented Generation (RAG) can also load the full script or policy text for the model to reference [7] [8].

- **Free-Form Conversation:** If there is no strict script, focus on expected behaviors (e.g. proper greeting, empathy). You might instruct the LLM to evaluate general compliance with guidelines. For example:

```
Prompt: “Analyze the following call transcript. Did the agent appropriately
greet the customer, verify their identity, and follow best-practice resolution
steps? List any best-practice elements that were missing.”
Transcript: {TRANSCRIPT}
```

This is less rigid but still checks for core elements. Use direct, specific language (see Compliance section below) to get clear answers.

Extracting Call Outcomes

Objective: Determine the result of the call (e.g. issue resolved, follow-up scheduled, no resolution).

- **Define Outcome Categories:** Decide the set of outcomes you want (e.g. “resolved”, “escalated”, “pending follow-up”). Clarify them in your prompt or schema.

- **Focused Prompting:** Ask the model directly for the outcome. For example, Regal AI recommends a prompt like **“What was the result of this call?”** to emphasize resolution [9]. A sample template:

```
Prompt: “You are a call summarization assistant. Based on the transcript,
clearly state the outcome of the call (e.g. ‘issue resolved’, ‘schedule follow-
up with agent’, ‘sale closed’, etc.) and give a brief explanation.”
Transcript: {TRANSCRIPT}
```

This steers the model to answer with the outcome. You can refine it (e.g. “In one sentence” or “Include any scheduled next steps”).

- **Structured Extraction:** Alternatively, extract outcomes via structured prompts. For instance:

```
Prompt: “Analyze the transcript. Output a JSON with keys:
  - `outcome`: a short description of the call result (“Resolved”, “Escalated”,
etc.),
  - `follow_up_required`: true/false,
  - `reason`: explanation if follow-up is needed.
Use the following transcript: {TRANSCRIPT}”
```

This JSON format ensures consistency. See the Snowflake example of asking for a JSON summary [10] [11].

- **Verification:** Always verify the model’s claimed outcome against the transcript (e.g. if it says “resolved”, check for phrases like “issue is resolved” in the text). Incomplete or incorrect outcomes can be caught by prompting the model to cite evidence:

```
“Which sentence in the transcript supports your determination of the
outcome?”
```

Analyzing Sentiment and Emotional Tone

Objective: Gauge customer sentiment (positive/negative) and emotional tone throughout the call.

- **Sentiment Classification:** Directly ask for overall sentiment. E.g.:

```
Prompt: “Overall, was the customer’s sentiment in this call positive,
negative, or neutral? Provide one sentence supporting your classification.”
Transcript: {TRANSCRIPT}
```

LLMs readily classify emotion (e.g. “The customer’s tone was frustrated and negative”). Glyph AI notes that LLM workflows often explicitly include *“Detecting Sentiment: Gauge emotions—positive, negative, neutral.”* [12].

- **Emotional Tone Analysis:** For deeper tone, ask about adjectives or style. For example:

```
Prompt: “Describe the customer’s emotional tone and language in this call.
Use terms like ‘angry’, ‘impatient’, ‘calm’, etc., and cite examples from
the transcript.”
Transcript: {TRANSCRIPT}
```

MarinSoftware suggests prompts like *“Describe the tone and language the customer uses… Is it formal, casual, technical, emotional, etc.?”* [13]. You can combine sentiment with tone for nuance (e.g. “frustrated but polite”).

- **Multi-Turn Emotions:** If you want sentiment per segment, split the transcript (e.g. per speaker turn or per topic) and analyze each part. Then aggregate. This can reveal shifts (e.g. sentiment before vs after resolution).

- **Advanced Tags:** Optionally, score specific emotions or use a scale (e.g. "On a scale of 1–5, how upset is the customer?"). However, for reliability, it's usually safer to stick to broad categories or qualitative labels.

Script Adherence and Compliance

Objective: Check whether the agent followed required procedures, language, and escalation flows.

- **List Key Script Elements:** Identify the mandatory parts of the script (greetings, disclaimers, specific phrasings, escalation triggers). Provide these as guidance. For example, Twilio's guide uses a prompt with bullet points:

  > *"Check for: Greeting & Identification – Did the agent introduce themselves?; Disclosure Statement – Did the agent provide the mandatory disclosure?; Resolution Steps – Did the agent follow the prescribed resolution process?; Closing – Did the agent summarize and offer help?"* [14] .

- **Yes/No Questions:** Ask direct questions about compliance. Twilio's example prompt begins with *"Review the transcript and evaluate whether the agent adhered to the required script. Check for the presence of key statements: ..."* [14] . Note how they break it into specific yes/no items. This directness boosts accuracy – Simon Greenman found that phrasing questions precisely raised compliance-check accuracy from ~69% to ~99% [15] .

- **Structured Output (JSON):** For reliable results, have the model return structured flags. In Twilio's sample, the output schema is JSON with boolean fields (e.g. `"greeting": true/false, "disclosure_statements": true/false, ...`) and a `score` plus `score_reason` [16] . You can similarly instruct:

  ```
  Prompt: "Analyze the following transcript against the company script.
  Output JSON with fields:
    - `greeting_ok`: boolean (was greeting used?),
    - `disclosure_ok`: boolean,
    - `resolution_steps_ok`: boolean,
    - `closing_ok`: boolean,
    - `score`: a numerical compliance score,
    - `score_reason`: brief explanation for score.
  Transcript: {TRANSCRIPT}"
  ```

  Ensuring a strict JSON schema (function calling or JSON mode) helps parse and validate the response [17] .

- **Retrieval of Policies:** If specific rules must be checked (e.g. “agent must not claim the test is free”), use retrieval to inject those rules. Genezio recommends building a knowledge base of internal docs/policies so the LLM stays grounded [8] . You could retrieve relevant policy snippets and include them in the prompt for compliance checks.

- **RAG and Chain-of-Thought:** For complex compliance (many rules), consider a multi-step chain: first retrieve relevant policy texts, then ask the LLM to compare transcript to those texts.

- **Prompt Engineering Tips:** As with general compliance, keep each prompt focused. Greenman’s team found it best to ask one question per prompt when possible [15] . For example, you might run separate prompts for each compliance check if needed, or use a single structured prompt like the JSON example above.

Summarizing Transcripts into Actionable Insights

Objective: Condense long transcripts into key takeaways (themes, customer pain points, resolution steps, etc.).

- **Key Themes/Pain Points:** Ask the LLM to list recurring issues. Example Marin prompt:

  ```
  Prompt: “Analyze this transcript and summarize the top 3 customer pain
  points, 3 objections, and 3 motivators mentioned. List them with brief
  explanations.”
  Transcript: {TRANSCRIPT}
  ```

  This gives a thematic summary, similar to MarinSoftware’s recommendation [18] .

- **Timeline/Bullet Summary:** For step-by-step notes, instruct bullet points. The Snowflake example tells the LLM to “be concise and use dot point format” for summarization [19] . For example:

  ```
  Prompt: “Summarize the call in bullet points: include the customer’s
  initial issue, each major step or question, and the final outcome.”
  Transcript: {TRANSCRIPT}
  ```

- **Structured Summaries (JSON):** As with script adherence, use JSON to capture categories. In Snowflake’s Cortex AI case, the prompt extracted fields like `initial_goal` , `primary_needs` , `current_issues` , `enthusiasm` , etc. from a long sales call [10] [11] . A similar JSON approach can ensure consistency. For instance:

  ```
  Prompt: “Extract the following from the transcript as JSON fields:
    - `customer_issue`: main problem customer had,
    - `agent_response`: how the agent addressed it,
    - `resolution`: final outcome,
  ```

```
  - `action_items`: list of any agreed next steps.”
Transcript: {TRANSCRIPT}
```

- **Dealing with Context Limits:** If the call is very long, you may summarize chunks first and then combine. For example: break the transcript into sections, run summarization prompts on each, then feed those summaries into one final prompt for an overall summary.

Recommending Coaching and Operational Changes

Objective: From the analysis, suggest targeted improvements for agents and processes.

- **Coaching Tips:** Use insights about agent performance to craft feedback. For example, if the LLM notes negative sentiment or missed greetings, your prompt can ask:

  ```
  “Based on the conversation, suggest coaching points for the agent (e.g.
  areas to improve on communication or compliance).”
  ```

  Or more systematically:

  ```
  Prompt: “Generate feedback for the agent on this call. Highlight what they
  did well and what could be improved (tone, script compliance,
  knowledge).”
  Transcript: {TRANSCRIPT}
  ```

  Glyph AI explicitly includes an “Agent Feedback” step in their workflow, aimed at “assessing performance and suggesting coaching” [20] .

- **Trend Analysis:** If multiple calls reveal the same issues, identify them for operational fixes. For example, ask the LLM to aggregate themes across calls (if processing in batch) or highlight recurring problems. These trends can inform training needs or script revisions.

- **Recommendations:** Convert findings into actionable changes. E.g. if customers are upset about shipping delays, recommend product team address logistics. If agents often skip parts of script, suggest changing training or script wording.
  You might use prompts like:

  ```
  “From this transcript, suggest any process improvements or additional
  training that could prevent similar issues in future.”
  Transcript: {TRANSCRIPT}
  ```

- **Performance Dashboards:** Ultimately, route these insights into dashboards or reports. The Snowflake use-case even automated sending summaries to Salesforce so agents see context [21] .

This shows how LLM outputs (like summarized notes or coaching tips) can feed back into the workflow for strategic use.

Sample Prompt Templates

Below are **prompt templates** illustrating how to frame each task. Replace placeholders (e.g. `{TRANSCRIPT}`, `{SCRIPT}`) with actual data.

- **Call Outcome Extraction:**

```
Prompt: "You are an AI assistant. Given the following call transcript,
determine the outcome:
  - Was the customer's issue resolved? (yes/no)
  - If no, what next steps were planned?
Provide a brief explanation for each answer."
Transcript: {TRANSCRIPT}
```

Output: JSON or bullet answers (e.g. `{"resolved": true, "explanation": "Order issue resolved by refund"}`).

- **Customer Sentiment:**

```
Prompt: "Analyze the emotional tone of the customer in this conversation.
Classify their overall sentiment as Positive/Neutral/Negative and explain
why with evidence from the transcript."
Transcript: {TRANSCRIPT}
```

- **Script Compliance (JSON):**

```
Prompt: "Check compliance with the script. For each required element below,
output true/false and a brief note:
  - Greeting (Agent introduced themselves?)
  - Mandatory disclosure (Agent gave required statement?)
  - Resolution procedure (Agent followed steps correctly?)
  - Closing (Agent summarized and offered help?)
Format the answer as JSON with keys: greeting_ok, disclosure_ok,
resolution_ok, closing_ok, notes."
Transcript: {TRANSCRIPT}
Script Elements: [Greeting, Disclosure, Resolution Steps, Closing]
```

- **Issue Summarization:**

```
Prompt: "Summarize the key points of this call in bullet points, including
the customer's main concern, how the agent responded, and the resolution or
next steps."
Transcript: {TRANSCRIPT}
```

- **Pain Points and Objections:**

```
Prompt: "Identify up to 3 customer pain points and 2 objections mentioned
in this call. List each with a short description and relevant quote from
the transcript."
Transcript: {TRANSCRIPT}
```

- **Agent Feedback Suggestion:**

```
Prompt: "Provide feedback for the agent on this call. Highlight at least
one thing the agent did well and one improvement point (e.g. tone, clarity,
script adherence)."
Transcript: {TRANSCRIPT}
```

Each template can be adjusted (e.g., asking for JSON output vs. plain text). You can also chain prompts, e.g. first extract a summary, then ask the LLM to analyze that summary further.

Diagnostics and Quality Control

To ensure accurate outputs and catch errors:

- **Automated Monitoring:** Track for common failure signs: formatting errors, irrelevant answers, hallucinations. As one guide advises, **“spot the problem”** by monitoring for vague responses or unsafe content [22]. Log LLM responses alongside prompts to analyze patterns.

- **Validation Rules:** Use function-calling or JSON schemas to enforce output structure [17]. Mismatches (malformed JSON, missing fields) can be auto-detected. For numerical or boolean outputs, verify they meet expectations (e.g. sentiment is one of {positive, neutral, negative}).

- **Cross-Checking:** Whenever possible, cross-check critical outputs. For example, if the LLM says “issue resolved,” scan the transcript for keywords (“resolved”, “closed”, “scheduled”) to confirm. You can even automate a second-pass prompt:

```
"Earlier you said the issue was resolved. Quote the part of the transcript
that shows the resolution."
```

- **Prompt Tuning:** If outputs are incorrect or incomplete, revisit the prompt. The **Latitudeblog** suggests iterating on prompts: simplify instructions, be more explicit, or break tasks into smaller ones [22]. For example, if a multi-part prompt fails, split it into separate questions.

- **Few-Shot Examples:** Provide one or two examples in the prompt if accuracy is low. Greenman's study found that using a few-shot (with examples of Q&A) plus RAG on smaller chunks gave big accuracy gains (sometimes +50% on specific tasks) [23].

- **Human-in-the-Loop:** Periodically have human agents review LLM outputs. Create feedback loops: if an output is wrong, use that as a learning example to refine the prompt or add to fine-tuning data. Genezio recommends running audits (manually reviewing flagged calls) or continuous monitoring to catch and fix mistakes early [24].

- **Safety and Compliance Checks:** Especially for compliance outputs, guard against LLM hallucinations. Use the transcription itself or external policy docs to confirm the LLM's assessment. Tools can highlight where the AI "missed the mark" in the conversation [24].

Retrieval Augmentation and Function Calling

For advanced tasks, consider these techniques:

- **Retrieval-Augmented Generation (RAG):** When prompts need external knowledge (e.g. product details or extended scripts), use RAG. Store your knowledge base (policy documents, FAQ, product manuals) in a retriever. At prompt time, retrieve relevant snippets and include them to ground the LLM's response. This is useful for compliance (fetch the exact rule text) or for detailed product info. One case found RAG had minor accuracy improvements overall, but drastically cut API usage by ~80% when splitting calls into chunks [23]. Moreover, RAG gave huge gains on few-shot tasks.

- **Function Calling / JSON Mode:** Modern LLMs allow declaring expected output schemas. If you have a defined data model (e.g. compliance flags, summary fields), use function calling or strict JSON mode. LlamaIndex notes that function calling is particularly effective for structured extraction when a schema is defined [17]. This ensures the LLM returns parsable JSON (with typed fields). For example, defining a function signature in OpenAI's API for extracting sentiment or outcome will produce a structured result you can directly use in code.

- **Embeddings & Vector Search:** For open-ended insights, you can embed transcripts and run semantic searches (via Pinecone or similar) to find relevant past calls or topics. Then prompt the LLM with those search results to enrich analysis. This can improve consistency and recall of historical patterns.

Recommendations and Best Practices

- **Iterative Refinement:** Continuously refine prompts based on feedback. Track metrics (accuracy of sentiment, compliance checks) and optimize. Small prompt wording changes (e.g. direct phrasing) can yield big accuracy gains [15].

- **Combined Strategies:** Use a mix of structured prompts (for compliance, JSON outputs) and open-ended prompts (for themes, coaching). For critical tasks (compliance, outcome), prefer structured schema outputs with explicit instructions [16] [17].

- **Human Oversight:** LLMs assist analysts, not replace them. Especially for compliance, always have a human review edge-case calls. Use AI to flag and summarize, but final judgment on sensitive issues should involve a person.

- **Data Privacy:** Always anonymize transcripts and remove PII before analysis. This is both a legal requirement and improves model performance on relevant content. (This is standard practice for any call analysis.)

- **Monitoring and Feedback Loop:** Implement dashboards or logs to track LLM performance. Set up periodic audits: Genezio recommends running one-time or ongoing audits, producing reports that highlight where the model "drifted off-topic" or violated rules [24]. Use these reports to adjust prompts or policies.

- **Scale with Care:** For large volumes, batch process transcripts but keep context constraints in mind. If context windows grow (e.g. using bigger models or multiple passes), ensure you're not hitting limits unexpectedly. Consider summarizing transcripts first to reduce length.

Overall, a successful LLM-based transcript analysis pipeline combines **clear prompt design, robust preprocessing, and systematic validation**. By following these guidelines – using direct, structured prompts and monitoring outputs – you can extract outcomes, sentiment, compliance scores, summaries, and coaching tips from calls at scale. Each insight (e.g. a recurring complaint or a skipped script step) can then inform coaching sessions or operational changes, turning raw transcripts into strategic action.

Sources: Industry guides and case studies on LLM call analysis [1] [2] [15] [24] [10] [14] [19].

[1] Analyzing Call Transcripts with LLMs
https://www.refuel.ai/blog-posts/analyzing-call-transcripts-using-llms

[2] [6] [7] [12] [20] How LLMs are used to extract insights from support calls
https://www.joinglyph.com/blog/how-llms-are-used-to-extract-insights-from-support-calls

[3] [15] [23] Best practices for LLM optimization for call and message compliance: prompt engineering, RAG, and fine-tuning | by Simon Greenman | Medium
https://medium.com/@sgreenman/best-practices-for-llm-optimization-for-call-and-message-compliance-prompt-engineering-rag-and-45ccca32ff17

[4] [5] [10] [11] [19] [21] Transforming +40min call transcripts into structured JSON summaries using mistral-large in Snowflake Cortex AI | by Chuang Zhu | Snowflake Builders Blog: Data Engineers, App Developers, AI/ML, & Data Science | Medium
https://medium.com/snowflake/transforming-40min-call-transcripts-into-structured-json-summaries-using-mistral-large-in-9b1c77af7377

[8] [24] Genezio | LLM Hallucination Detection for AI Agents in Customer Service
https://genezio.com/deployment-platform/blog/llm-hallucination-detection/

[9] 6 Use Cases for REGAL AI Call Summaries
https://www.regal.ai/blog/six-use-cases-for-regal-ai-call-summaries

[13] [18] Unlocking the Voice of the Customer: Using LLMs to Elevate Your Marketing with Sales Call Transcripts | Marin Blog
https://www.marinsoftware.com/blog/unlocking-the-voice-of-the-customer-using-llms-to-elevate-your-marketing-with-sales-call-transcripts

[14] [16] Generative Custom Operators | Twilio
https://www.twilio.com/docs/conversational-intelligence/generative-custom-operators

[17] OpenAI JSON Mode vs. Function Calling for Data Extraction - LlamaIndex
https://docs.llamaindex.ai/en/stable/examples/llm/openai_json_vs_function_calling/

[22] 5 Steps to Handle LLM Output Failures
https://latitude-blog.ghost.io/blog/5-steps-to-handle-llm-output-failures/