Task 1: Strategic Design of Chatbot Analytics Framework

Example: (1) Develop a customer support chatbot

In a retail banking environment, where customers seek quick resolutions for queries like account balances, transactions, or loan applications, a comprehensive analytics strategy is essential for optimizing chatbot performance.

⇨ This strategy includes performance metrics, user interaction logging, and business key performance indicators (KPIs), justified by their contribution to innovation in evaluation.

Key Components of the Strategy

1. Chatbot Performance Metrics:

- **Intent Recognition Accuracy**: Measures the percentage of user intents correctly identified, using precision, recall, and F1-score. For banking, high accuracy prevents misrouting queries (e.g., confusing "balance check" with "transfer funds").

- **Response Time:** Average time from user input to chatbot reply, targeting under 2 seconds to maintain engagement.

- **Fallback Rate:** Proportion of interactions referred to human agents, ideally below 10%, indicating robustness.

2. User Interaction Logging:

- Log all sessions with timestamps, user IDs (anonymized for privacy), intents, entities, and conversation flows. Use structured formats like JSON for easy querying.

➔ This enables funnel analysis.

What is Funnel Analysis?

Funnel analysis is a method used to track how users move through a predefined sequence of steps (a "funnel") toward a goal — like completing a task in a chatbot.

Example in a chatbot:

Funnel:

1. User greets the bot
2. Selects "Book an appointment"
3. Enters preferred time
4. Confirms booking

Funnel analysis result:

- **Step 1:** 1000 users
- **Step 2:** 800 users
- **Step 3:** 500 users
- **Step 4:** 300 users

Insight: Major drop-off at Step 3 → maybe the date/time input UI is unclear or too slow.

How is funnel analysis calculated in Rasa?

Funnel analysis in **Rasa** for a chatbot allows you to evaluate **how users progress through predefined conversation stages**, helping to identify where users **drop off** in the flow (e.g., onboarding, FAQ, booking, etc.).

Step-by-Step Funnel Analysis in Rasa

- ❖ **Step 1: Define the Funnel Steps**

These are key checkpoints in your chatbot conversation.

Example:

Funnel: "Product Inquiry → Add to Cart → Checkout → Confirmation"

In Rasa terms, these are usually:

- **Intents**
- **Actions**
- **Custom events (slots, forms, etc.)**

❖ **Step 2: Extract Conversation Logs**

You need access to the conversation data. This is typically stored in:

- **Rasa Tracker Store** (PostgreSQL, MongoDB, Redis, etc.)
- **Rasa X** exports
- **Custom telemetry logs** (e.g., from a database or data lake)

---

3. Business KPIs:

- **Conversion Rate**: Percentage of interactions leading to actions like fund transfers or account openings, linking chatbot efficacy to revenue.

- **Customer Retention Rate**: Measured by repeat interactions within 30 days, reducing churn through personalized follow-ups.

- **Cost Savings**: ROI calculated as (human agent hours saved * hourly cost) / development cost, optimizing resource allocation.

ROI: stands for Return on Investment

In the context of chatbots, ROI measures the financial benefit gained from using the chatbot compared to its cost.

$$\textbf{ROI (\%)} = \frac{\text{Total Benefits} - \text{Total Costs}}{\text{Total Costs}} \times 100$$

Example in chatbot analytics:

Costs:

- **Development**: €10,000
- **Hosting & maintenance**: €2,000/year
- **Total cost** = 10,000+2,000= €12,000

Benefits (per year):

- Rerouting 4,000 support calls
- Each call costs €5 (if handled by human agent)
- **Total savings** = €20,000

$$\mathrm{ROI} = \frac{20,000 - 12,000}{12,000} \times 100 = \mathbf{66.7\%}$$

So, you gained **66.7% return** on your initial investment.

Why It Matters in Chatbots

- Measure cost-efficiency of deploying a chatbot (e.g., support savings vs development cost).
- Justify AI projects to stakeholders.
- Optimize performance (e.g., high ROI = better business value).

Selected Analytics Types and Justification

- **A/B Testing**: Deploy two chatbot versions (e.g., one with rule-based responses vs. LLM-enhanced) to compare metrics like conversion rates. This fosters innovation by validating improvements, such as a 15-20% uplift in engagement from personalized prompts.

- **Funnel Analysis:** Visualizes user journeys from greeting to resolution, identifying bottlenecks (e.g., high drop-offs in KYC verification). It contributes to performance evaluation by enabling data-driven redesigns, like simplifying flows.

- **Cohort Analysis:** Groups users by demographics (e.g., age, account type) to track retention, supporting adaptive models that personalize responses (e.g., simpler language for elderly users).

Task 2: Research and Critique of Industry Optimization Approaches

What you'll deliver: For each sector you chose:

(1) a case-study summary,

(2) the KPI list you'll target,

(3) the dataset(s) and synthetic log plan,

(4) your analytics events + proposed dashboards, and

(5) an A/B test design (hypothesis, primary metric, sample allocation, runtime).

Then, Compare these approaches with current trends such as adaptive dialog flow models, multivariate testing, and prompt engineering for LLMs.

Task 3 — Practical Implementation Strategy

0) Pick your base chatbot (2–5 minutes)

- **Option A — Your own bot** from last semester (Rasa, Dialogflow, Botpress).
- **Option B — Ready-made**: e.g., a Rasa example bot or a Botpress blueprint/demo. Official docs for each platform are fine to cite. Rasa+1Botpress

1. Session Heatmaps

Session heatmaps visualize **user navigation paths** and **time spent** at each step in a chatbot conversation.

> ➢ This helps identify which nodes (intents, actions, or topics) are most engaging or where users tend to drop off.

Example Visualization:

- Sankey diagram or heatmap showing transitions between intents/actions
- Color gradient to show time-on-node (dark = long dwell time)

Tools:

- Python + Plotly / Dash / Matplotlib
- Rasa tracker data processed to extract sequence paths

2. User Segmentation & Personalization

This involves **grouping users based on their behavior** or attributes (e.g., intent frequency, preferred channels, language used) and then tailoring responses or experiences to each segment.

Segmentation Criteria:

- **Intent frequency** (e.g., user A asks about shipping, user B about pricing)
- **Channel usage** (e.g., WhatsApp vs web users)
- **User profile** (e.g., location, login status, language)

Tools:

- Python + Pandas for segment identification
- Dashboards with Plotly or Seaborn
- Rasa custom actions or slot-based conditions for personalization

3. Accessibility or Fallback Optimization Techniques

This involves improving the chatbot's **ability to handle misunderstandings or communication barriers**, ensuring it is inclusive and can handle errors.

Techniques:

- **Fallback strategy tuning**: Adjust fallback_threshold, use multiple fallback actions (action_default_fallback, action_ask_rephrase)
- **Rephrasing prompts**: Ask user to reword unclear input
- **Multi-language support**: Add translations or language detection
- **Accessibility design**: Simple buttons, speech-to-text integration, screen-reader friendly formatting

Tools:

- Rasa fallback policy tuning
- Language detection libraries (e.g., langdetect)
- Accessibility testing tools (e.g., Lighthouse for web UI)

Task 4: Critical Evaluation and Testing Strategy

A robust evaluation strategy for the banking chatbot includes **A/B testing**, **statistical testing for dialogs**, and **dialogue anomaly detection**, supporting user-centric improvements.

1. **A/B Testing:** Test variants (e.g., **Version A:** Standard responses; **Version B:** Personalized via LLMs) on metrics like conversion rate.
   - Randomly assign 50% users to each, using t-tests for significance.
2. **Statistical Testing for Dialogs:** Apply statistical methods to assess whether observed improvements or differences (e.g., user satisfaction, retention) are **statistically significant**.

   Example:

   - Chi-square test on fallback rates between two versions
   - t-test on average conversation length or resolution time

3. **Dialogue Anomaly/Intent Drift Detection:**

Definition: Monitoring for **unexpected dialogue patterns** or **changes in user intent usage** over time, which can indicate:

- Broken conversation flows
- Emergence of new user needs
- Intent misclassification

Techniques:

- Anomaly detection using clustering (e.g., DBSCAN)
- Intent frequency monitoring

- Embedding-based similarity drift tracking (e.g., sentence-transformers)

User-Centric Benefit:

- Enables **early detection of issues** users face
- Supports **adaptive design** as user needs evolve

Task 5: Insightful Reporting and Visualization

Design a **dashboard** that presents clear, actionable insights from chatbot usage data—helping both technical and non-technical stakeholders understand how well the chatbot is performing and where it can improve.

Dashboard Requirements Explained

1. Cross-Platform Performance

Show how the chatbot performs across different channels (e.g., web, WhatsApp, Facebook, Telegram).

Key Metrics:

- Number of sessions per channel
- Average satisfaction score per platform
- Completion rate by channel

Visualization Ideas:

- **Bar chart:** Sessions per channel
- **Pie chart:** Platform usage distribution
- **Line graph:** Daily interactions by platform

---

2. User Journey Attribution

Track how users move through key conversation stages (i.e., the funnel), and where they drop off.

Metrics to Include:

- Step-by-step conversion funnel (e.g., Greeting → Info Request → Booking)

- Drop-off rates at each step
- Time spent per step

Visualization Ideas:

- Sankey diagram for user flows
- Funnel chart for drop-off analysis
- Heatmap of intent transitions

---

3. Feedback / Implicit Signals

Capture user feedback and non-verbal cues (implicit signals) such as:

- **Manual ratings (1–5 stars):** Explicit user ratings at the end of a session (e.g., “How satisfied are you with this chat?”)
- **Intent confidence scores:** The confidence level (between 0 and 1) with which the NLU model classifies an intent.
- **Fallback frequency:** The number of times a fallback action (e.g., action_default_fallback) is triggered due to misunderstanding or low intent confidence.
- **Negative sentiment detection:** Identifying emotional tone or dissatisfaction in user messages using **sentiment analysis**.
- **Abandonment rates:** Percentage of users who start a session but exit before completing the expected journey (e.g., without reaching booking_confirmation).

Visualization Ideas:

- **Donut chart:** Feedback scores
- **Heatmap:** Confidence scores across intents
- **Bar chart:** Top fallback-triggering intents

Tools You May Use

- **Plotly Dash** or **Streamlit** for interactive dashboards (Python-based)
- **Google Colab** for running simulations or data analysis
- **Flask + Seaborn/Matplotlib** for static dashboards
- **Metabase / Tableau** (if using a real DB and UI integration)