**CIS 96L NLP - Professors Sanjay Dorairaj & Tuan Nguyen**

***Community Pulse Feasibility Study by Hung Lu & Jesse Katz***

### ***1. Project Overview***

***Title: Community Pulse: Automated Sentiment Analysis from Reddit for East Bay Civic Insights***

***Description: Community Pulse is a sentiment analysis and visualization tool powered by Natural Language Processing (NLP) techniques. It continuously scrapes posts and comments from region-specific Reddit communities (e.g., r/oakland, r/eastbay) to analyze public discourse and track emerging trends in community sentiment. It will offer a user-friendly dashboard to display sentiment over time, categorize key issues, and trigger alerts about noteworthy shifts in tone or topic frequency.***

***Objectives:***

* ***Automate the collection of textual data from Reddit's East Bay community forums.***
* ***Preprocess and analyze the data using NLP techniques.***
* ***Visualize trends and categorize emerging community concerns.***
* ***Provide accessible insights to users in government, education, nonprofits, and community advocacy.***
* ***Support user-specified alerts and geospatial insights using maps.***
* ***Offer comprehensive visualizations such as time-series sentiment charts, word clouds, topic frequency bars, and heat maps.***
* ***Allow users to define filters, date ranges, keywords, and alert conditions via an interactive dashboard.***

***Success Criteria:***

* ***Accurate and timely data extraction (daily or hourly scraping).***
* ***85%+ accuracy in sentiment classification (validated via human labeling).***
* ***Functioning dashboard with real-time updates, visual filters, and export tools.***
* ***Ability to detect and categorize new issues with keyword clustering.***
* ***User-triggered alerts and map-based trend exploration.***
* ***Effective visualization of data patterns using multiple chart formats.***

***Target Users:***

* ***Local government agencies***
* ***Neighborhood organizations***
* ***Activist groups and nonprofits***
* ***Concerned citizens and researchers***

***Expected Impact:***

* ***Real-time awareness of local concerns***
* ***Better-informed community interventions***
* ***Support for proactive policymaking and civic engagement***

### ***2. Data Acquisition & Processing Analysis***

***Data Sources:***

* ***Reddit subreddits: r/oakland, r/eastbay, r/BayArea, and others***
* ***Backup/expanded sources: Publicly available police blotters, city council meeting transcripts, local college campus alerts (e.g., emergency notifications or student forums), 311 service request feeds, community blogs, and public Facebook or Twitter data from official agencies (with proper compliance).***
* ***Data Diversity: NextDoor or local Facebook groups to capture sentiment from demographics that may not use Reddit***

***Access Method:***

* ***Reddit API (Pushshift for historical data, Reddit’s official API for real-time scraping)***
* ***Web scraping for public websites and alerts (BeautifulSoup, requests)***
* ***Python-based scraper with PRAW and requests libraries***

***Legal & Ethical Considerations:***

* ***Complies with Reddit API’s rate limits and terms of service***
* ***Only public posts and comments will be analyzed***
* ***Personally identifiable information will be stripped***
* ***Backup sources will be used only if public and within fair-use/data-sharing terms***
* ***User Privacy: Detailed anonymization process for handling public posts***

***Data Storage:***

* ***PostgreSQL or MongoDB for flexible querying and text indexing***
* ***AWS S3 or local disk for raw logs and backups***
* ***Optional cloud storage: Google Drive (via PyDrive or Google Drive API) for smaller or shared data components***
* ***Historical Context: Storage plan for historical data collection to establish baseline sentiment patterns***

***Preprocessing Needs:***

* ***Tokenization, stopword removal, lemmatization***
* ***Sentiment labeling using pre-trained transformer models (e.g., BERT)***
* ***Named Entity Recognition (NER) and topic modeling for issue identification***
* ***Multilingual Analysis: Processing for Spanish and Chinese content to represent East Bay's diversity***

***Sample Exploration Results (preliminary):***

* ***A sample dataset of 1,000 posts from r/oakland revealed frequent topics around housing, crime, and local events. Initial sentiment tagging using VADER showed 70% neutral, 20% negative, and 10% positive sentiment.***

***Optional User Controls:***

* ***Users will be able to specify a custom date range through the dashboard interface before data is scraped. This will ensure efficient storage usage and prevent over-collection of unnecessary data.***
* ***Users can subscribe to email or SMS alerts based on custom criteria (e.g., spikes in negative sentiment about crime).***

### ***3. Technical Feasibility Assessment***

***NLP Techniques:***

* ***Sentiment analysis using fine-tuned transformer models (BERT, RoBERTa)***
* ***Keyword extraction (TF-IDF, RAKE)***
* ***Named Entity Recognition***
* ***Topic modeling (LDA, BERTopic)***
* ***Trend detection and event clustering***
* ***Custom Training: Models trained on local data to better capture regional slang/terms***
* ***Trend Prediction: Predictive component to forecast emerging community issues***

***Hardware/Software Requirements:***

* ***Mid-to-high-end local machine (16GB+ RAM) or cloud VM (AWS EC2 or Google Cloud)***
* ***Python environment (Anaconda recommended)***

***Tools & Libraries:***

* ***Scraping: PRAW, Pushshift API, BeautifulSoup***
* ***NLP: spaCy, Hugging Face Transformers, NLTK***
* ***Dashboard: Streamlit, Plotly, Flask (for API)***
* ***Alerts: Twilio or SendGrid API for notifications***
* ***Mapping: Folium, Leaflet.js, or Mapbox for geographic visualization***

***Team Competency:***

* ***Team members have experience in Python, REST APIs, NLP techniques, and basic data engineering.***

***Limitations & Solutions:***

* ***Reddit rate limits: Use asynchronous batch jobs and rotating tokens.***
* ***Sarcasm and irony detection: Incorporate contextual transformers like RoBERTa.***
* ***Topic drift: Refresh keyword clusters weekly.***
* ***Location inference from text can be imprecise: supplement with NER and manual geo-tagged keywords.***

### ***4. Implementation Plan***

***Timeline (6 Weeks):***

* ***Week 1: Setup and Reddit API access + initial data collection script***
* ***Week 2: Preprocessing pipeline and basic sentiment tagging***
* ***Week 3: Build dashboard skeleton, connect live data***
* ***Week 4: Advanced NLP integration (NER, topic modeling, alerts)***
* ***Week 5: Testing, map overlay integration, UI polish***
* ***Week 6: Final presentation + documentation***

***Deployment: Phased approach starting with core functionality***

***Tasks & Responsibilities:***

* ***Data Collection:***
* ***NLP Pipeline:***
* ***Dashboard Development:***
* ***Alert System & Mapping:***

***Methodology:***

* ***Agile with weekly deliverables and sprint reviews***

***Testing & Validation:***

* ***Manual validation of sentiment and entity detection***
* ***A/B testing different classifiers***
* ***Alert test with simulated triggers***
* ***Validation Strategy: Defined process for human validation of sentiment accuracy***

***Deployment Strategy:***

* ***Local machine deployment first, then optional deployment to Heroku or AWS***

### ***5. Risk Assessment***

***Technical Challenges:***

* ***Sentiment misclassification***
* ***Insufficient Reddit data volume during quieter weeks***
* ***Mapping inaccuracies from inferred location data***

***Data Risks:***

* ***Reddit content removal or subreddit restrictions***
* ***Legal access concerns with backup sources if not properly vetted***

***Time Management Risks:***

* ***Overlapping coursework deadlines***
* ***Scope Concerns: 6-week timeline ambitious given NLP task complexity***

***Mitigation Strategies:***

* ***Use cached data and fallbacks***
* ***Set buffer weeks in project plan***
* ***Limit scope to essential features if time runs short***
* ***Test mapping module on known geo-tagged data first***

***Contingency Plans:***

* ***Shift to analyzing Twitter, local alerts, or Nextdoor if Reddit becomes inaccessible***
* ***Replace real-time scraping with static dataset analysis***
* ***Delay geographic visualizations if location data proves unreliable***

### ***6. Resource Requirements***

***Compute Resources:***

* ***Local PC (16GB RAM, SSD) or Google Colab Pro***
* ***Optional AWS EC2 or Google Cloud VM for deployment***

***Storage Requirements:***

* ***Daily data: ~5MB/day; Monthly: ~150MB***
* ***PostgreSQL DB or MongoDB Atlas (~1GB total estimate)***
* ***Cloud storage option: Google Drive for shared projects and CSV exports***

***External Services:***

* ***Reddit API (free)***
* ***Optional: AWS (~$10/month for low-tier VM)***
* ***Google Drive API (free tier sufficient for prototype)***
* ***Twilio (for SMS alerts, $0.0075/message) or SendGrid (free for basic email tier)***
* ***Mapbox (free tier for visualization up to 50,000 map views/month)***

***Time Commitment:***

* ***6–8 hours per week over 6 weeks***

### ***7. Proof of Concept***

***Core Functionality:***

* ***Reddit scraper in Python using PRAW and Pushshift***
* ***NLP preprocessing with spaCy and Hugging Face***
* ***Sentiment analysis with pre-trained transformer***

***Results:***

* ***Prototype script collected 1,000+ posts from r/oakland***
* ***Sample sentiment tagging achieved 82% match with human labels***

***Code Snippets:***

**# Initialize a sentiment analysis pipeline using Hugging Face Transformers,**

**# analyze the sentiment of a sample sentence about local governance,**

**# and output the predicted sentiment label and confidence score.**

from transformers import pipeline

sentiment\_model = pipeline("sentiment-analysis")

result = sentiment\_model("The city council really failed us this time.")

print(result)

**# Text Preprocessing example**

def preprocess\_text(text):

"""Clean and preprocess text data"""

if not isinstance(text, str):

return ""

text = text.lower()

text = re.sub(r'http\S+', '', text)

text = re.sub(r'[^a-zA-Z\s]', '', text)

tokens = word\_tokenize(text)

cleaned\_tokens = [lemmatizer.lemmatize(token) for token in tokens if token not in stop\_words]

return ' '.join(cleaned\_tokens)

**# Sentiment Analysis Function**

def analyze\_sentiment(text):

"""Analyze sentiment of text using transformer model"""

if not text or text.strip() == "":

return {"label": "neutral", "score": 0.5}

try:

result = sentiment\_model(text[:512]) # Truncate to avoid token limit

return result[0]

except Exception as e:

print(f"Error in sentiment analysis: {e}")

return {"label": "neutral", "score": 0.5}

### ***8. Enhanced Visualization & UX Features***

***User Customization:***

* ***Filtering by neighborhood within cities (e.g., different Oakland neighborhoods)***
* ***Comparison Features: Ability to compare sentiment across different geographic areas***

***Temporal Analysis:***

* ***Day/night pattern analysis to identify time-specific concerns***
* ***Weekend vs. weekday sentiment comparison***

***Accessibility:***

* ***Ensure dashboard is accessible to users with disabilities***
* ***High-contrast mode and screen reader compatibility***

***User-Centered Design:***

* ***Involve potential end-users in UI design process***
* ***Iterative testing with target user groups***

***Prepared by: Jesse Katz & Hung Lu Date: April 4, 2025***