**ユーザビリティ 最終課題 アプリケーション「Gyaha」仕様書**

K20001 相畑亮太

1. **はじめに**

昨今では，機械学習によるデータ分析技術が飛躍的に発達している．機械学習によって実現された技術は多種多様に渡り，画像に対して「どの位置に何が写っているか」を分析する物体検知技術や，テキストデータを入力として音声を合成する音声合成技術などが具体例として挙げられる．  
　機械学習を利用した技術の一つに，ダジャレの判定が挙げられる．ダジャレ判定技術は，入力された文字列がダジャレかどうかを判定することができる．河合(2020)によると，機械学習を利用したダジャレ判定手法として，DajaRecognizerが挙げられる[1]．

これらの技術はあくまでダジャレの判定手法でしかなく，実際に機械学習モデルやAIが人間の代わりに笑ってくれる訳ではない．しかし，いくつかの技術を組み合わせることにより，ダジャレを言うと何らかのリアクションを返してくれるアプリケーションを実装することは可能である．今回のユーザビリティの最終課題に対する題材として，以上の要件を満たすアプリケーションを制作した．

1. **ダジャレを聞いて笑ってくれる人工知能「Gyaha」**「Gyaha」は，ダジャレを言うと，何らかのリアクションを返してくれるアプリケーションである．アプリ名は「ギャハハハ」という笑い声から命名した．  
   　Gyahaの大まかな流れを説明する．初めにダジャレをマイクから入力し，日本語の文字列に変換する．この処理には音声認識の技術を利用した．次に，入力された文字列がダジャレかどうか判定する．ダジャレの判定には，ダジャレ判定アルゴリズム「Shareka\_v4」を利用した．最後に，ダジャレの判定結果に応じて，音声を再生したり，画面を変化したりさせ，リアクションを表現する．リアクションの音声には，音声合成技術を利用し生成した笑い声を利用している．  
   　Gyahaには「ダジャレを聞いて笑ってくれる人工知能」というテーマが設定されている．このテーマには，ダジャレを実際に人工知能に聞いてもらっているようなUXを提供するという意味が含まれている．そのため，アプリケーション内でそのようなUXを提供するための工夫を凝らした．例えば，ユーザにダジャレをテキスト入力してもらうのではなく，実際に言ってもらう仕様にしたのも，このUXを提供することが目的である．他にも，人工知能感を演出するために，わざとダミーの計算時間を用意したり，一定のタイミングで合成音声を使用してユーザに語りかけたりするように実装した．
2. グラフィカル ユーザー インターフェイス, アプリケーション

   自動的に生成された説明**Gyahaの画面リスト**　Gyahaを起動すると以下のような画面が表示される．

図 1: Gyaha起動時の画面

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明　スタートボタンを押すと，音声認識画面に移行する．

図 2: 音声認識画面

　この画面に移行する際に，アプリケーションから「ダジャレを言ってください」と合成音声で呼びかけられる．ユーザーは指示に従ってダジャレ音声を入力する．

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明　なお，音声認識に失敗した場合は，音声合成によるメッセージと共に以下のような画面に遷移する．

図 3: 音声認識エラー画面

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明　音声認識に成功した場合，以下の画面に遷移する．

図 4: 判定中画面

この画面に遷移した瞬間に，ユーザが入力したダジャレが画面上に表示され，「ピロピロ」という計算中であることを思わせるような効果音が再生される．また，以下の画面に遷移した瞬間にダジャレの判定処理が実行される．ダジャレの判定が終わり，かつ判定開始してから最低5秒経過した場合，次の画面に遷移する．5秒経過させる目的は，ユーザに「ピロピロ」という効果音を聞かせることによって「計算している感」を演出し，人工知能に触れているようなUXをユーザに与えることである．

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明先に述べた条件を満たすと，以下の画面に遷移する．

図 5: 判定結果画面

判定結果画面では，大きな笑い声とファンファーレの音声が再生される．画面に表示されていた顔は笑顔になり，ユーザは「人工知能にダジャレを笑ってもらった」というUXを獲得する．

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明なお，入力された音声がダジャレじゃないと判定された場合，以下の画面に遷移する．

図 6: 入力された音声がダジャレじゃないと判定された場合の判定結果画面

判定結果画面下部の「もう一度」ボタンを押すことによって，音声認識画面に遷移し，もう一度Gyahaにダジャレを聞いてもらうことができる．

1. **Gyahaのプログラムソースについて**プログラムソースに関する補足事項を述べる．VoiceRecorder.pyは，音声認識を行うためのスクリプトである．特に工夫した点は無く，ライブラリSpeechRecognionを利用して授業内スクリプトと同じように実装を行った．

SoundPlayer.pyは，wavファイルを再生するためのスクリプトである．主にライブラリPyAudioを利用して実装した． Streamを利用することによって，複数のwavファイルを同時に再生する処理を実現することができた．再生中の音声を途中で止めることも可能である．また，SoundNameRepository.pyは，音声ファイルのコードとパスを対応付けるスクリプトである．実装は辞書型で行ったが，万が一実装が変更になっても良いように，クラスでカプセル化を行い隠蔽した．また，LaughtPlayer.pyは，合成音声の笑い声5種類を全て再生するクリプトである．合成音声は，先に録音し保存しておくことで，実行する度に合成するよりも高速に実行できるようにした．

　Shareka\_v4.pyは，文字列がダジャレかどうかを判定するためのスクリプトである．このスクリプトは私が全て作成したものではなく，vaaaaanquish氏のスクリプトを改造したものである．元々動作しないスクリプトであったが，数行変更することによって動作を可能にした．必要なライブラリはMeCab, alkana, pykakasi, fuzzysearchである．また，mecab専用の辞書であるmecab-ipadic-neologdも利用している．FunnyUtil.pyは，Shareka\_v4を呼び出し，ダジャレであれば1, そうでなければ0を返すスクリプトである．つまり，FunnyUtilはShareka\_v4をラップしたクラスである．ダジャレ判定アルゴリズムを変更する際はFunnyUtilを書き換えるだけで良く，他のスクリプトの変更は不要である．  
　UI.pyはライブラリPySimpleGUIによるUIを構成し，UIを変更するインタフェースを提供するスクリプトである．ボタンを押した際の処理を外部から受け取るように実装することによって，クラスUIを完全にロジックから切り離した．GUIライブラリを変更する際は，UI.pyを書き換えるだけで良く，他のスクリプトの変更は不要である．  
　MainProcessは，以上のスクリプトを利用し，全体のプロセスを制御するクラスである．ボタンを押された後の処理はスレッドとして実行することによって，PySimpleGUIのループを止めないようにした．

1. **実行方法**zipをダウンロードしたら解凍し，展開されたディレクトリでターミナルを起動し，以下のコマンドを入力する．  
   $ brew install python-tk@3.9 portaudio mecab mecab-ipadic  
   $ pip3 install -r requirements.txt  
   　以上が環境構築のコマンドである．環境を構築した後は  
   $ ./launch  
   　と入力し，launchを実行する．
2. **まとめ**　いつの時代でも，ダジャレを言って，反応がもらえると嬉しいものである．Gyahaはいくらでもユーザのダジャレを聞いて，笑ってくれる人工知能アプリケーションである．将来的には，Gyahaのようなアプリケーションを応用した「ダジャレ版カラオケ」のような，ダジャレを言うためのお店も現れるかもしれない．Gyahaは，そんなアイディアを初歩的に実現した新しいアプリケーションである．
3. **参考文献**

[1] 河合 俊典(2020)「ダジャレを判定する」(<https://vaaaaaanquish.hatenablog.com/entry/2020/12/11/122721>) 検索日 2022/01/25