

卒論の目次

1 序論

- 1.1 近年の機械学習
 - 1.1.1 人工無能
 - 1.1.2 人工知能
 - 1.1.3 neural network
 - 1.1.4 DeepLearning
- 1.2 一般的な人工知能開発フレームワーク
 - 1.2.1 Chainer
 - 1.2.2 DL4J
 - 1.2.3 TensorFlow
- 1.3 一般的な人工知能の返答の流れ

2 提案

- 2.1 現在実在するフレームワークの問題点
 - 2.1.1 人工知能との会話のビジュアル
 - 2.1.2 入出力部分の作成
 - 2.1.3 その他の人工知能との連携
- 2.2 人工知能ハブの提案
 - 2.2.1 全体構成
 - 2.2.2 アルゴリズムのみを簡単に追加可能な知能ハブ
 - 2.2.3 作成したアルゴリズムをすぐにUnityで試せる機構
 - 2.2.4 Unityが利用可能なモーションを追加する機構

3 設計

- 3.1 解析アルゴリズムの追加を可能にする機構
 - 3.1.1 解析したい内容別にプログラムを保持する機能
 - 3.1.2 会話の話題別に解析するプログラムを選ぶ機能
 - 3.1.3 解析アルゴリズムを簡単に追加する機能
- 3.2 解析済み情報を共有する機構
 - 3.2.1 解析情報を保存する機能
 - 3.2.2 解析済みの情報を取得する機能
- 3.3 返答アルゴリズムの追加を可能にする機構
 - 3.3.1 出力を行うタイミング
 - 3.3.2 出力する内容別にプログラムを保持する機能
 - 3.3.3 会話の話題別に解析するプログラムを選ぶ機能
 - 3.3.4 返答アルゴリズムを簡単に追加する機能
- 3.4 Unityで作成した知能を試せる機構
 - 3.4.1 藤井さんの論文を引用してゴニョゴニョ言う

- 3.4.2 Unityとの連携に利用するWebsocket
- 3.4.3 Unityへの送信フォーマットの作成
- 3.4.4 Unityからの受信フォーマット
- 3.5 Unityが利用可能なモーションを追加する機構
 - 3.5.1 鈴木氏の論文を引用してゴニョゴニョする
 - 3.5.2 MongoDBからデータを取得する機構
- 3.6 アルゴリズムを選定する際に用いるGoogleAPI
 - 3.6.1 GoogleAPIについて
 - 3.6.2 GoogleAPIの有効性

4 実装

- 4.1 開発環境
 - 4.1.1 Javaの利用
 - 4.1.2 Mavenフレームワーク
- 4.2 解析部分の実装
 - 4.2.1 解析分野別にアルゴリズムを保持する機構
 - 4.2.2 会話の話題別に解析するアルゴリズムを選択する機構
 - 4.2.3 解析アルゴリズムを3行で簡単に追加する機構
 - 4.2.4 現在実装している解析アルゴリズム
- 4.3 データベースの実装
 - 4.3.1 全ての解析情報を保存する機構
 - 4.3.2 解析した情報を取得する機構
- 4.4 出力情報を作成する機構
 - 4.4.1 出力分野別にアルゴリズムを保持する機構
 - 4.4.2 会話の話題別に出力を作成するアルゴリズムを選択する機構
 - 4.4.3 返答アルゴリズムを3行で簡単に追加する機構
 - 4.4.4 現在実装している出力アルゴリズム
- 4.5 Unityとの通信の実装
 - 4.5.1 Unityからの入力情報の受信
 - 4.5.2 Unityへの命令の送信
- 4.6 追加したモーションの利用
 - 4.6.1 動作選択アルゴリズムの実装
- 4.7 GoogleAPIと形態素解析を用いた頻出単語表の作成する機構
 - 4.7.1 形態素解析による検索ワードの作成
 - 4.7.2 GoogleAPIを利用して検索結果を取得
 - 4.7.3 検索結果のフィルタリング
 - 4.7.4 頻出単語表の作成

5 実行結果

- 5.1 Unityの出力画面の図
- 5.2 実際の会話
- 5.3 アルゴリズムを追加した後の会話

6 結論

- 6.1 アルゴリズムの追加による出力の変化
- 6.2 Googleを用いた会話の話題推定の精度
- 6.3 簡単にアルゴリズムを追加できたか

7 参考文献

メモ

設計の部分ではコンセプトを伝える(図や画像を使う)

実装の部分ではソースコードを用いて丁寧に実装がどうなっているのかについて解説する

水野さんメモ

- ・ 1章の近年の人工知能を機械学習にしたほうがしっくり来る
- ・ DeepLearningの話を入れるならニューラルネットワークについても入れる
- ・ 一般的な人工知能開発フレームワークにテンソルフローを入れよう
- ・ 提案、設計、実装は同じような内容で良いけど同じような流れで説明すること
- ・ ちゃんと提案 設計 実装 の内容を分別して記述すること
- ・ 実装のUnityとの通信のWebsocketとかは実際3章の設計にあたる(実装をしていないから)
 - ー送信する情報はこういう設計になっていてすぐに送れるとか
 - ー受信する情報はこういう設計になっていてすぐに解析できるとかは3の設計に当たる
- ・ 同じくGoogleApiを利用するということを伝えるのも3章の設計に当たるはず
 - ーだけど頻出単語表を作成する部分は実装したよね
 - ー形態素解析とか
 - ーそれは実装のプログラムの部分に書かないとだめぽ
- ・ 実装の部分でソースコードレベルで書くもの
 - ー3行でアルゴリズムを追加できるとか
 - ーインターフェイスはこうなるとか

3章の全体の設計の部分はどちらかと言うと設計というよりも提案のところかな

今の2.2のところにこういう全体の設計で行きますよって書くことになる。
コレは多分書いていたら気づくことだけどねって行ってた

保存

解析アルゴリズムのみの開発
解析済み情報の共有
対話アルゴリズムのみの開発
Unityとブラウザによる入出力の完備