エッジコンピューティングを活用したスマートスピーカー

2018SE026 加瀬翔大
指導教員：宮澤元

1　はじめに

近年，世界においてスマートスピーカーが広く普及するようになった [1]．スマートスピーカーとは，ユーザーの音声による指令に反応して様々な処理を行う IoT 機器である．ユーザーの音声指令でスマート家電を制御したり，インターネットを介してユーザーが欲しい情報を伝えたりすることができる．

スマートスピーカーの問題点としてクラウドで処理を行うことによる応答遅延とプライバシー漏洩が指摘されている．スマートスピーカーによる音声処理はクラウドで行われるので，音声指令などの情報をクラウドとの間で送受信するのに時間がかかり，応答時間が長くなってしまう．また，クラウドサーバーに保存された音声情報が不正アクセス等で流出した場合，ユーザーが話したプライバシーに関わる内容が第三者に知られてしまう恐れがある．

これらの問題を解決する方法としてエッジコンピューティングが提案されている．エッジコンピューティングとは，デバイスそのもの，あるいはデバイスに近い位置に配置したサーバーでデータの処理を行うコンピューティング技術である．クラウドにおけるデータ処理やネットワーク負荷を軽減したり，デバイスから低遅延でサービスにアクセスしたりすることができる．また，プライバシーを含む情報をデバイスの近くで処理してインターネットを介してやり取りしないようにすることで，プライバシー情報の漏洩の問題を軽減できる可能性もある．

本研究の目的は，エッジコンピューティングを活用するスマートスピーカーを実現することである．一般にデバイスやエッジサーバーの計算リソースはクラウドサーバーほど潤沢ではないので，これらの機器の計算リソースがスマートスピーカーの処理を行うのに十分かどうか確認する必要がある．

本稿では，エッジコンピューティングを活用するスマートスピーカーの構成について提案する．スマートスピーカーシステムをマイクを持つ IoT デバイスであるスマートスピーカー本体とエッジサーバーおよびクラウドサーバーから構成する．ユーザーからの音声指令に対する音声認識と形態素解析をスマートスピーカー本体とエッジサーバーで行い，指令内容に応じて残りの処理をエッジサーバーで行うかクラウドサーバーで行うか決定する．エッジ側で処理すべき内容としては，ホームネットワーク上の機器の操作やプライバシー情報を含む要求などがある．処理内容に応じてエッジとクラウドを使い分けてデータ処理を行うことにより，処理の応答時間の改善とプライバシー漏洩問題の軽減を両立できる．

研究課題は，スマートスピーカー機能の提案と実装を行うことと，音声認識を行う場所や通信遅延，プライバシー情報の扱いについて提案手法の妥当性の検討を行うことである．

2　スマートスピーカーの構成

一般的なスマートスピーカーシステムは，Smart Speaker と Cloud Service からなる (図 1)．Cloud Service は，Smart Speaker からの要求を受けて処理を行う Front End と，Home Network の制御を行う Back End に分かれる．

Smart Speaker はユーザーがマイクに発言した内容を受け取ると，それを Front End Cloud Service に伝送し，そこで音声認識が行われる．また，外部のサービスから必要な情報を Front End Cloud Service が受け取り，Smart Speaker に Front End Cloud Service がレスポンスを返すことで，Smart Speaker が必要な情報を音声化して出力する [2]．

家庭の IoT 機器をスマートスピーカーで起動させたい場合は，Front End Cloud Service から Back End Cloud Service に音声データを伝送する．そして，Back End Cloud Service で IoT 機器の種類と IoT 機器の場所をリストアップし，Home Network に命令を伝送することで，IoT 機器を起動させることができる．

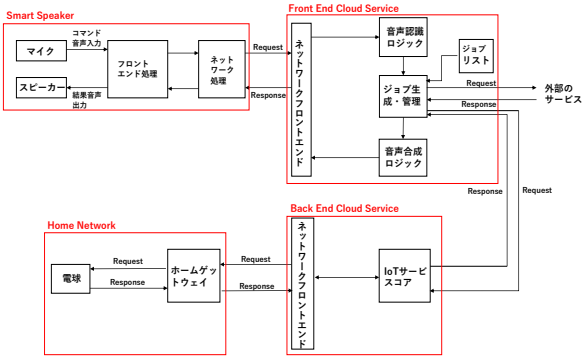

図 1　スマートスピーカーのシステム構成 [2]

3　提案手法

提案するスマートスピーカーシステムの構成を図 2 に示す．Smart Speaker 内にユーザーが音声を発音するマイクが搭載されている．ユーザーが発音した音声を認識するシステムである音声認識ロジックの処理を Smart Speaker 側で行う．形態素解析器は音声認識ロジックと接続されている．また，形態素解析器はクラウドサーバーとエッジサーバーと接続されていて，条件に応じて処理を行うサーバーを選択する．Home Network の処理もしくはプライバシーに関わる内容など，エッジ側だけで完結させられる処

理であればエッジサーバーで処理を行い，そうでなければクラウドサーバーで処理を行う．クラウドサーバーとエッジサーバーはいずれも Smart Speaker 内のフロントエンド処理と接続されているので，ユーザーが発音した内容に応じて結果を返答する．

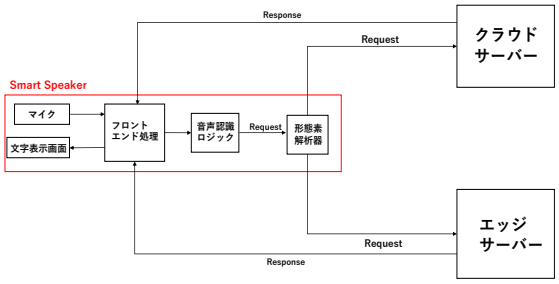

図 2　提案するスマートスピーカーシステムの構成

4　提案するスマートスピーカーの実装

MacBook Air を用いて Smart Speaker を，Windows PC を用いてエッジサーバーを実現した．クラウドサーバーは Amazon Web Services(AWS) 上で動作させた．音声認識には CMUSphinx[3] を，形態素解析には TreeTagger[4] をそれぞれ使用して実装している．

形態素解析結果がプライバシー情報を含む場合はエッジサーバーに，そうでなければクラウドサーバーに送信する．今回は，形態素解析結果が人名を含むかどうかでプライバシー情報を判定することとした．

今回はエッジサーバー及びクラウドサーバーからは単純な文字列を返信するだけで，スマートスピーカーの具体的な処理は実装しなかった．

5　実験

実験の目的は，エッジサーバーでデータの送受信を行う場合とクラウドサーバーでデータの送受信を行う場合で，通信反応時間にどのような違いがあるのかを確かめることである．

5.1　実験結果

CMUSphinx の処理時間，TreeTagger の処理時間，クライアントプログラムとサーバー間による送受信の処理時間をそれぞれ 15 回計測し，データの平均値と全体の処理時間 (合計値) を**表 1**，**表 2** に算出した．

実験結果から，クライアントプログラムとエッジサーバー間の送受信の処理時間の方がクライアントプログラムとクラウドサーバー間の送受信の処理時間よりも 0.19 秒早いことがわかった．

また，CMUSphinx による音声認識処理がエッジサーバーに対して送受信を行う場合の処理時間の約 84% を占めており，かなり負荷の高い処理であることがわかった．

表 1　エッジサーバーを利用した場合の処理時間 (s)

利用システム	user	system	cpu(%)	total
CMUSphinx	1.81	0.11	64.3	3.05
TreeTagger	0.46	0.04	102.8	0.48
エッジサーバー	0.06	0.03	78.2	0.11
合計	2.33	0.18		3.64

表 2　クラウドサーバーを利用した場合の処理時間 (s)

利用システム	user	system	cpu(%)	total
CMUSphinx	1.81	0.11	64.3	3.05
TreeTagger	0.46	0.04	102.8	0.48
クラウドサーバー	0.08	0.03	35.7	0.3
合計	2.35	0.18		3.83

6　おわりに

本稿ではエッジコンピューティングを活用したスマートスピーカーについて述べた．クラウドサーバーで送受信を行う時間とエッジサーバーで送受信を行う時間を測定した実験の結果，エッジサーバーで送受信を行う方が通信応答時間が 0.19 秒早かった．エッジサーバーに対して送受信を行う場合の処理時間の内訳の約 84% を音声認識処理が占めており，音声認識をスマートスピーカーで行うと負荷が高すぎる可能性がある．また，提案手法におけるプライバシー保護について確認する実験等を行うことができなかった．

今後は，実際に利用者の発話に対する処理を行うことができるスマートスピーカーを実装した上で，提案手法の有効性を確認する．

参考文献

[1] 総務省：Iot デバイスの急速な普及,https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/html/nd111200.html (2023/01/10 access).

[2] INTERNET Watch，呼び掛けにどう応答しているのか？ Amazon Echo や Google Home が動く仕組み，https://internet.watch.impress.co.jp/docs/column/nettech/1107574.html (2023/01/17 access).

[3] CMUSphinx Open Source Speech recognition, https://cmusphinx.github.io/ (2023/01/10 access).

[4] TreeTagger - a part-of-speech tagger for many languages, https://cis.uni-muenchen.de/ schmid/tools/TreeTagger/ (2023/01/10 access).