VAEを用いた画像圧縮，異常検知，（題名もうちょっと良いのないかな）

今村　優希[†a)]　　川崎　大雅[†b)]

Yuki IMAMURA[†a)] and Taiga KAWASAKI[†b)]

あらまし　ああ（大体 300 文字）

キーワード　VAE, FPGA, エッジコンピューティング

1. はじめに

近年，無線通信技術は飛躍的に向上しており，5G 通信の普及が進んでいる．5G は従来の 4G など通信規格と異なり，「高速大容量」「低遅延」「多数同時接続」の 3 つの特徴を備えており，その中でも「低遅延」と「多数同時接続」は新たな通信環境を構築する上で重要な軸となっている [1].

従来の 4G 通信は，人が使用するスマートフォンや携帯に焦点を当てていた．しかし，5G では車両，ドローン，センサなどの IoT 機器が大量にネットワークに接続されることを前提としている．

このような環境においては，従来のクラウド中心の処理方式では，トラヒック増加による遅延や負荷集中が生じる可能性があり，5G の利点を十分に発揮できない問題がある．また，今現在の IT 業界ではクラウドが主流で，処理の多くを一つ（もしくは複数の）コンピュータで行うという構造である．多くの端末から取得したデータをクラウドのみで処理を行うのはある程度限界があり，またトラヒック量が増加して，5G のメリットを享受できないという問題が発生すると考えられる．

このような課題を解決するため，エッジコンピューティングという技術が近年注目され始めている．エッジコンピューティングとは，従来はクラウドで行っていた処理の一部を，ユーザ端末（スマートフォンや IoT 機器）の近い位置である基地局やその至近に設置されているサーバなどでデータ処理を行う技術である [2]. この技術を用いることでクラウドにかかる処理をエッジコンピューティングで分散することが可能で，通信のトラヒック量，5G の特徴のひとつである「低遅延」に貢献することも可能である．

そこで，エッジコンピューティングの実現を VAE と FPGA を用いて実現することを考えた．(本文を書きながら続きを記述予定)

2. 方　　法

2.1 実行環境

今回の設計において使用したツール及びそのバージョンを表 1 で示す．

2.2 システム構造

今回の VAE を搭載したシステムでは，以下 2 つの

† 九州工業大学情報工学部 情報・通信工学科 3 年
福岡県飯塚市川津 680-4
Kyushu Institute of Technology, School of Computer Science and System Engineering, Department of Computer Science and Networks
a) E-mail: imamura.yuuki475@mail.kyutech.jp
b) E-mail: kawasaki.taiga000@mail.kyutech.jp

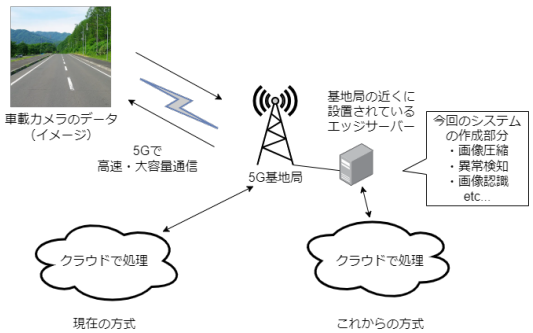


図 1: エッジコンピューティングのイメージと
今回のシステムの作成部分

表 1: 使用したツール

用途	使用ツール
VAE シミュレーション用	MATLAB 2024b
ハードウェアシミュレーション	MATLAB/Simulink 2022a
HDL コード生成	HDL Coder
FPGA 設計ソフトウェア	Vivado 2022.1
ハードウェアアクセラレーション	Vitis xxxx
評価ボード	DIGILENT 製 ZYNQ-7010

機能を搭載した．

1　画像圧縮

VAE の特徴のひとつである次元圧縮能力を画像に応用する．

2　異常検知

もう一つの特徴である異常検知を，元画像と生成画像とを比較して行う．

全体の構想について解説する．今回使用する画像は，簡単化のためにグレースケール化したものを使用する．また，JPEG のようにブロックに分割して，それぞれのブロック毎に処理を行う．ブロックサイズは 16×16 に設定した．

画像圧縮や異常検知の判定は，元画像と圧縮後の画像との比較を PSNR を用いて行う．PSNR が設定した閾値以上だった場合は精度良く圧縮ができるているので圧縮した潜在空間の値を送信する．それ以下だった場合は異常として通知を行う．

上記の機能を実現するために，VAE の構造を 2.2.1 で説明する．また，FPGA の構造の詳細を 2.2.2 にて説明し，最後に SoC FPGA の構造を 2.2.3 にて説明する．

2.2.1　VAE 構 造

VAE の構造の概略を図 2 に示す．16×16 の画像を使用することから，入力 256 次元，出力 256 次元で設

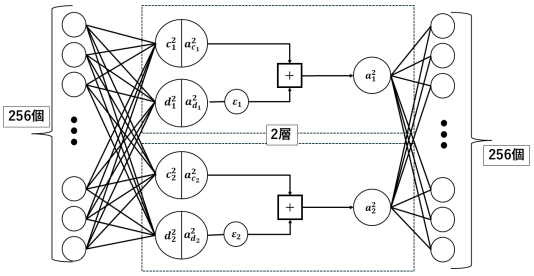


図 2: 今回の VAE の構造

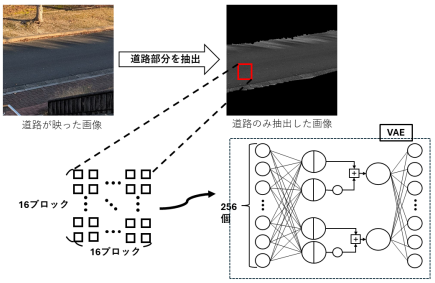


図 3: VAE 学習方法

計を行った．潜在空間の次元は，次元圧縮と異常検知という目的を両立させるために，16 次元で設計を行った．エンコーダ部分の活性化関数に関しては，平均では ReLU 関数を使用し，分散ではソフトプラス関数を使用している．また，デコーダ部分ではシグモイド関数を利用している．

$$f(x) = x \qquad : ReLU \text{ 関数} \qquad (1)$$

$$f(x) = \log(1 + e^{x}) \qquad : Softplus \text{ 関数} \qquad (2)$$

$$f(x) = \frac{1}{1 + e^{-x}} \qquad : Sigmoid \text{ 関数} \qquad (3)$$

また，VAE の学習方法を図 3 に示す．まず，道路のみが映った画像を用意する．その画像に対して，道路のみ映った部分だけで 16×16 のブロックにする．そのブロックらを教師データとし，VAE を学習させる．今回のシステムでは，MATLAB で VAE の学習のみ行い，そこから出力された重みやパラメータを使用して LSI 設計を行った．学習の条件を表 2 にまとめた．

表 2: 学習条件

epoch	10000
eta	0.0005
Layer2(潜在空間の数)	16

表 3: FPGA のリソース利用率

Resorce	Utilization	Available	Utilizatio[%]
LUT	3301	17600	18.76
LUTRAM	62	6000	1.03
FF	3297	35200	9.37
DSP	64	80	80.0
IO	12	100	12.0
BUFG	1	32	3.13

2.2.2 FPGA 構造

使用したボードは，DIGILENT 製の ZYNQ-7010 である．今回の設計では，入力が 34，出力が 2 の演算を行う．

入力データとして，X，$w2$，b を使用する．X と $w2$ は 16 個，b は 2 個である．赤色の枠で囲われているユニットは，

$$Output_1 = X_1 \times w2_1 \tag{4}$$

のような，入力と重みのパラメータを乗算する演算を行っている．その演算ユニットを 16 個用意したものが，青色の枠線で囲われているユニットである．青色のユニットでは，最終的に以下の計算を行っている．

$$Z_{21} = \sum_{i=0}^{16} X_j \times W_{2j} + b2_1 \tag{5}$$

そのユニットを 2 つ用意することで，出力を 2 つ得られるように設計した．

当初は入力 256，出力 256 の演算を FPGA に載せたかったが，容量に限界があった．したがって，今回設計した 256×16×256 の VAE に切りの良い数字を使った，X の入力が 16 になるように設計を行った．

FPGA に実装した後の評価を表 3 に示す．

2.2.3 SoC FPGA の構造

SoC FPGA のシステム構造の概要を図 5 にて示す．Processing System ユニットがバスを通って様々な処理を行う．

次に，SoC FPGA の処理の概要を図 6 にて示す．SD カードに格納されている VAE の学習重みや画像データを CPU が読み取る．その後，作成した FPGA に従って入力データの処理を行い，FPGA にそのデータを送信する．FPGA はデータが格納されると同時に実行し，出力結果を保存する．その出力結果を CPU が読み取りに行き，出力データを処理する．それらの処理を繰り返し行う．

今回の VAE が 256×16×256 であり，2.2.2 で作成

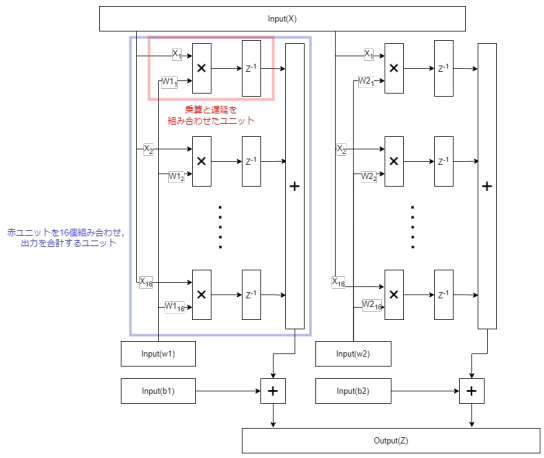


図 4: FPGA の構造

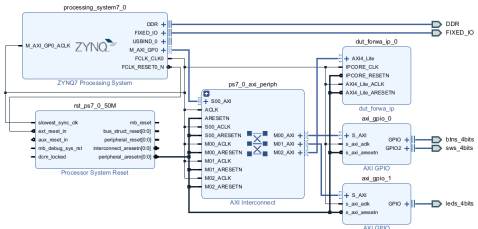


図 5: SoC FPGA の構成図

した FPGA の構造が 16×2 である．設計した FPGA でエンコーダを計算する際は，一つの出力で z^2_{mean} を，もう一つの出力で z^2_{var} を計算させる．1 つの潜在空間を計算させるためには，FPGA を 16 回使用する必要がある．その潜在空間が 16 個あるので，FPGA は合計で $16 \times 16 = 256$ 回稼働する．デコーダ部分を計算する際は，FPGA を一回稼働させるだけで，3 層目の出力を 2 つ得られる．したがって，デコーダでは 128 回 FPGA を稼働させる．このような制御を CPU で作成した．

2.3 実 験 方 法

今回は，図 7 のような車載カメラから取得した画像をイメージとしたものを利用する．また，プロトタイプの作成として，画像の一部のブロックに対して VAE を活用した画像圧縮，異常検知を行うようにした．一部のブロックの選出については図 7a のように設定しており，車がこれから通過するであろう部分を判定するように設定した．図 7a における赤枠の大きさは，16×16 である．

実際に，図のようなテスト画像を用意した．実際に

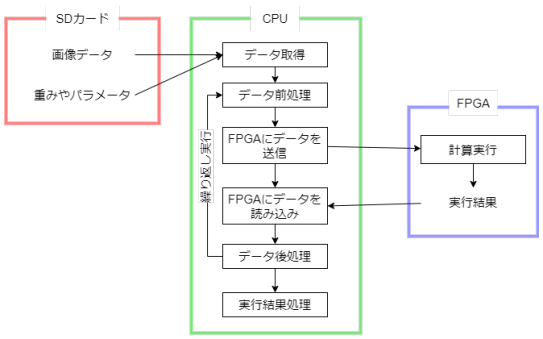
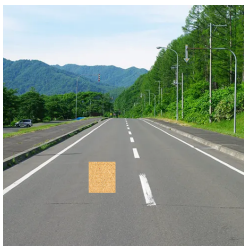


図 6: SoC FPGA の処理概要

(a) 車載カメラからのイメージ図

(b) 評価対象例
赤枠の部分を VAE にて評価する

図 7: 今回のシステムで評価する方法
これいらんかもな



(a) 評価する画像

(b) 評価画像をグレーに変更

図 8: 今回テストする評価画像

VAE で判定させるものは図の白黒画像である．この図は，道路に落下物が落ちていることを想定しており，今回のシステムにおける異常検知の実施を行う．

3. 実験と考察

3.1 実 験 方 法

3.2 実 験 結 果

えーまだ，この FPGA を使用したメインプログラムが作成できていないので記述できません...

3.3 考　　察

出力された画像とシミュレーションした画像の比較を行う．

4. 結論と今後の展望

結論として，

今回は，エッジコンピューティングを意識し，VAE を FPGA を用いて実装を行った．5G が浸透していき，様々な IoT 機器がネットに繋がるようになると考えられる．その際に，いかに効率よく伝送し，早く制御を行うかが重要になってくると思う．今後，VAE や FPGA が自動運転を含む様々な分野で応用されていくと考えられ，今回の開発はこれからの発展の初期的な内容のものであると考えることができる．

使用した評価ボートに載せることのできる回路に制限があり，思うような回路を作成することができなかった．しかし，16 入力-2 出力の回路を作成し，この回路で VAE のエンコードもデコードもできるようプログラムできたことは良かったと感じる．これからも FPGA を用いた開発を行っていきたいし，高位合成等の FPGA を開発するための技術も日々進化しているので，様々なことに挑戦していきたい．

謝辞　今回のシステム構築に対して，様々な支援を頂いた方々に感謝する．

文　　献

[1] 森川博之, 5G 次世代移動通信規格の可能性, 岩波書店,
[2] 田中裕也, 高橋紀之, 河村龍太郎, ”IoT 時代を拓くエッジコンピューティングの研究開発”, NTT 技法ジャーナル, vol.27, no.8, pp.59-63, 2015.