改良エッジパターン法による文書画像への電子透かし

三浦, 周平 / MIURA, Shuhei

(発行年 / Year)
2014-03-24
(学位授与年月日 / Date of Granted)
2014-03-24
(学位名 / Degree Name)
修士(工学)
(学位授与機関 / Degree Grantor)
法政大学 (Hosei University)

2013年度

修士論文

改良エッジパターン法による文書画像への電子透かし

指導教員

彌冨　仁　准教授

法政大学大学院　工学研究科　情報電子工学専攻修士課程

12R4149　三浦周平

目 次

第1章　はじめに

電子透かしとは、人の目には認識できないほどの微小の変化をデジタルコンテンツに施すことで情報を埋め込み、必要に応じて埋め込まれた情報を検出することである。電子透かし技術は、コンテンツに著作権情報を埋め込むことで著作権情報の保護や改ざんの検知を行う手段を提供している。

近年インターネットを利用し簡単にデジタルコンテンツが入手でき、品質も劣化しないままにコンテンツをコピーできるようになった。このことでたくさんの不正ダウンロードや配布が行われ著作権が侵害されることが多く問題となっている。不正行為が行われないよう著作権の保護や不正コピーの防止ができる技術が求められている。文書を画像とみなしたとき、それは白黒の２値画像を考えることができる。紙に印刷してしまった文書は再デジタル化は困難で、OCR（Optical Character Reader）技術も発達してきているものの誤認識も多い。もし文書画像に対してある程度多くの情報を適切に埋め込み、またそれらを誤りなく読み込みできれば紙とデジタルの融合が進み非常に有意義な技術になりうる。例えばの著作権を埋め込むことや、あるいは文書と同じ情報を埋め込むことができれば超高精度 OCR にもなり得る。

本研究では文書画像にエッジパターン法を応用した手法を用いて非可視性をある程度保ったまま情報を埋め込む方法について開発を行った。埋め込める情報量の増加をさせ、誤り訂正符号を付加して埋め込むことでロバスト性を向上させるとともに位置ずれの問題に対処する。また、ハフ変換を用いて画像中の文書から直線を複数検出しそれらをもとに傾きを求め、画像の非可視性を保ったままに回転によって引き起こされる位置ずれを解決することでロバストな電子透かし技術の確立を目標とする。

第2章　電子透かし

2.1　2値画像の電子透かし

文書画像を中心として二値画像は広く使われており、そのセキュリティの必要性が高まっている。画像データにおいては、コンピュータ上で扱われる際に JPEG に代表される圧縮やフィルター処理が行われるため、それらの処理が施されても透かし情報を残すことのできる技術が必要となってくるが、二値画像においては、ガンマ変換やエッジ強調・平滑化などのフィルター処理、JPEG に代表される不可逆圧縮などは通常行われないため、それらの処理に対する耐性は考慮しなくてよい。しかし二値画像は一画素を 1 ビット (白と黒) のみで表現するため、データサイズが小さくて済むが、画素値における情報の冗長性が極めて小さいため、非可視性を保ったままで情報の埋め込みを行うのは困難である。特に白、或いは黒のベタ領域の画素値を変更すると位相が変化するために不自然さが目立ってしまい、情報の埋まっていることがすぐに知られてしまう。二値画像に透かし情報を埋め込む手段としては、ファクシミリ画像などで用いられているランレングス符号に対して透かし情報を埋め込むものがある [1]。しかし、この方法は縦の直線などにジャギーが発生するなど画質の劣化が大きい。二値の文書画像に対する従来の電子透かし技術の手法は、行間や文字間の間隔を調整することで情報を埋め込む方法 [2] や、文書から抽出した特長量を透かし情報として背景地紋に埋め込む方法 [3]、エッジ部分を変更することで情報を埋め込む手法がある。本研究では、文書の各文字のエッジを利用したエッジパターン法 [4] を基礎として開発を行う。エッジパターン法は3× 3のウィンドウを利用し、そのウィンドウを画像左上から右下までスキャンさせ文書のエッジパターンを読み取り、微小な変化を与えることで「1」、「0」の bit 情報を埋め込む。以下に二値の文書画像に電子透かしを埋め込む方法の説明する。

2.1.1 行間調整法

行間調整法または行移動コーディング法（Line-shift Coding）は、基本的に文書における文字の行を行単位で移動することによって、行間の距離を平均距離より狭くさせたり、広くさせたりすることで、「1」か「0」の透かしのビット情報を文字の行間に埋め込む手法である。Fig.2.1 に示したのは行間調整法の一つの例だ。

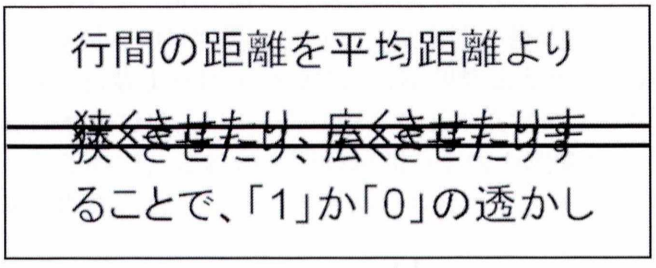


Fig. 2.1: 行間調整法の例

2.1.2 文字間調整法

文字間調整法または文字移動コーディング法 (Word-shift Coding) は、文書画像の文字間に情報を埋め込む方法である。元の文書画像を文字単位で文字間の距離を平均距離より狭くさせたり、広くさせたりすることによって「1」「0」の情報を文字間に埋め込む方法である。Fig.2.2 に示しているのは文字間調整法を用いたひとつの例である。上の行の文字「of」前のスペースは、下の行の文字「of」 前のスペースより広くなっていることがわかる。これが変更されていることに気付くのは容易ではない。文字間調整法のメリットとしては、通常、文書一頁における文字数は行数よりは多いので、行間調整法よりも大量のデータの埋め込みが可能となる。しかし日本語では、「い」や「川」などの文字のように、文字内に隙間がある文字がよく使われているので、これが字間の隙間か字内の隙間なのか判断が困難である。この判断を誤って情報を埋め込むと見た目がおかしくなってしまう。

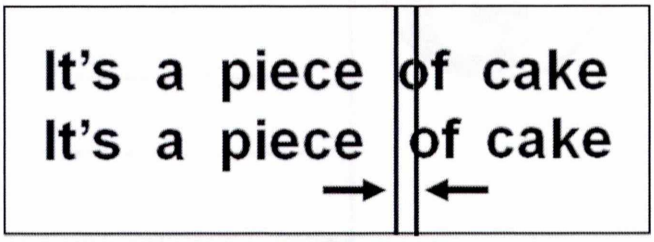


Fig. 2.2: 文字間調整法の例

2.1.3 背景地紋法

背景地紋法は、文書から抽出した特徴量を透かし情報として背景地紋に埋め込むことで、文書の改ざんを検知することが可能となる方法である。一方、背景地紋法では、スキャン画像から元の文書の特徴量の再抽出を、より高い正確率で保障できないと、改ざんを検出する際に誤った判断をしてしまうことがあり、文書の改ざんがないものを、改ざんされたものとして判断してしまう可能性がある。スキャン画像から元の文書の特徴量をもっと高い確率で抽出することが、今後の最大の課題である。Fig.2.3 に背景地紋の例を示す。

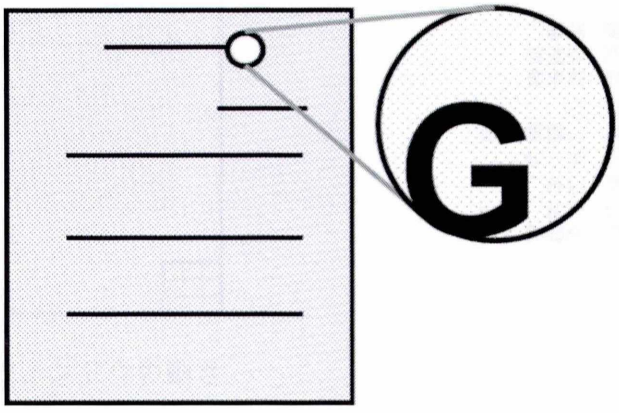


Fig. 2.3: 背景地紋法

2.1.4 エッジパターン法

これまでに提案されているエッジパターン法は、3× 3のウィンドウで画像中をFig.2.4のように重なりなくスキャンしていき、文字のエッジから検出したウィンドウパターンがFig.2.5の中にある予め定めた数種類のウィンドウパターンと一致した場合に「1」、「0」のbitを表すパターンに変更することで情報を埋め込む。変更ルールは定めたパターンの数だけあるがFig.2.6に変更ルールを示す。

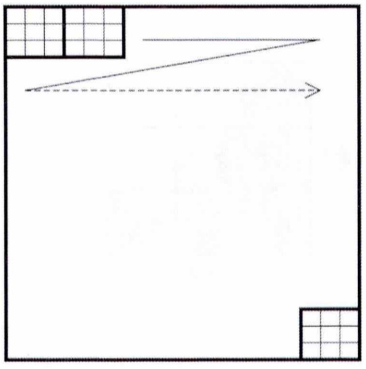


Fig. 2.4: ウィンドウの動き

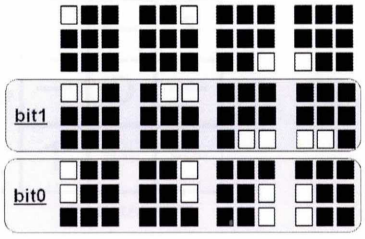


Fig. 2.5: 既存のエッジパターン

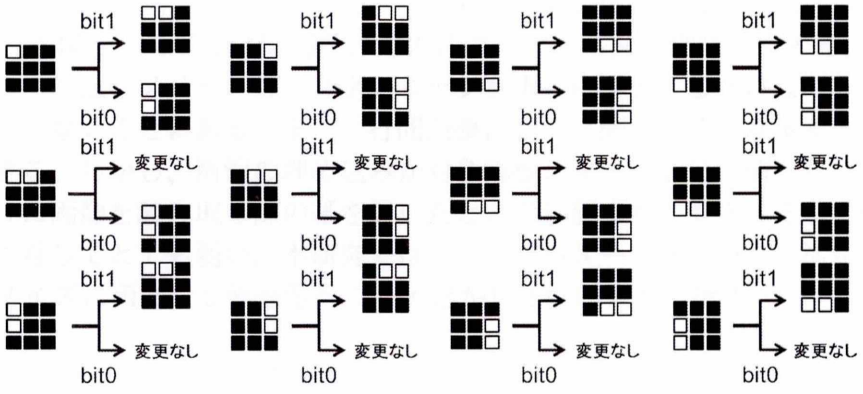


Fig. 2.6: 変更ルール

Fig.2.7 は以上に示したルールで埋め込み処理を模式的に行った例である。この例では、埋め込む情報を「101」とし、画像の左上からスキャンしていくと右上の最初のエッジ部分で Fig.2.5 の情報を持たないエッジパターンと一致するので変更ルールに沿って「1」を表すエッジパターンに変更する。次に中央のエッジが「1」に一致するパターンなので同様にルールに沿って「0」を表すエッジパターンに変更する。最後に左下のエッジが「1」のパターンと一致していることから変更しない。これらの処理により情報「101」が画像に埋め込まれたことになる。

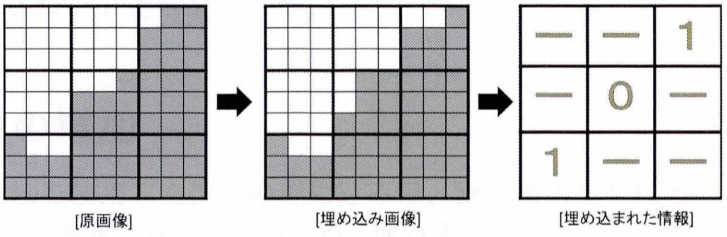


Fig. 2.7: 情報を埋め込んだ例

埋め込み情報を読み取るときは、エンコード時と同様に左上からスキャンしていき定義したエッジパターンの「1」「0」を表すパターンを検出するたびに情報を保存していく。これを繰り返し 8bit 保存するごとに区切り、ASCII コードから文字列に変換する。

エッジパターン法のメリットは、文字のエッジを微小に変更するだけなので劣化が少なくほとんど目立たないことにある。また、行間調整法や文字間調整法より多くの情報を埋め込むことができる。しかし、情報の埋め込みが対象画像のエッジ部分に依存しているため、スキャンのように画像を読み取る際の紙を置いたときの位置ずれや傾き、そして紙自体の拡大縮小、歪みに対してとても弱い。本研究では、エッジパターン法の利点を活かし欠点を改善することでノイズに頑健な 2 値画像への電子透かし法の確立を目指す。

第3章　エッジパターン法の改良

この章ではエッジパターン法の欠点である、埋め込み可能な情報量の少なさ、位置ずれ、傾きに対する方法について述べる。

3.1　情報量の増加

エッジパターン法を使用した情報の埋め込みは、画質の劣化を極力抑えるために文字のエッジ部分にのみ情報を埋め込む。そのため、埋め込むことのできる情報量がとても少ない。この欠点を改善するために、従来の 12 パターン以外に、埋め込みが可能な新たなるパターンを探す。新たなパターンを探す条件として、情報を埋め込むことによるエッジの変化が認知されにくい、つまり画像の劣化が抑えられるパターンであること、また数が多く情報量を増やせるパターンであることが挙げられる。この条件に当てはまるパターンの候補を挙げ、実際に透かし情報を埋め込みその情報を埋め込んだ画像と原画像の画質の劣化を比べ、実際に使用可能か調べる。以下に新たに提案するエッジパターンとその変更ルールを示す。

- パターンA

既存のエッジパターンを白黒反転させたエッジパターンをパターンAとする。Fig.3.1にエッジパターン12種を示し、Fig.3.2に変更ルールを示す。

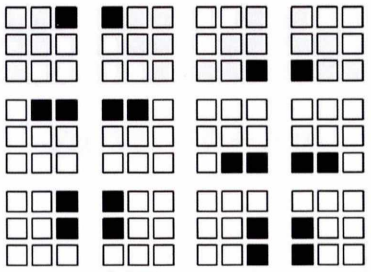


Fig. 3.1: パターンA

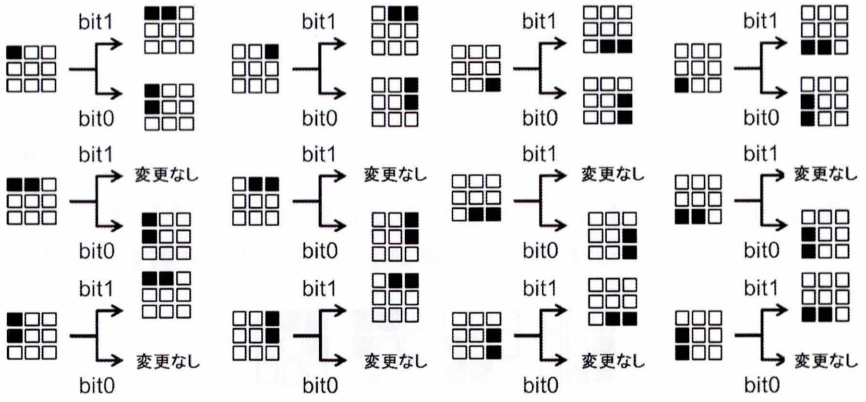


Fig. 3.2: パターンAの変更ルール

- パターンB

9画素の角3ピクセルが白いパターンをパターンBとする。Fig.3.3にエッジパターン4種を示し、Fig.3.4に変更ルールを示す。

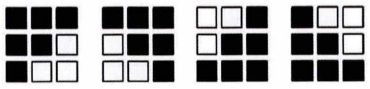


Fig. 3.3: パターンB

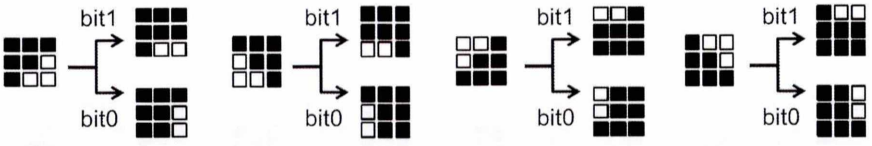


Fig. 3.4: パターンBの変更ルール

- パターンC

タイプB の4 パターンを白黒反転させたパターンをパターンC とする。Fig.3.5にエッジパターン4種を示し、Fig.3.6に変更ルールを示す。

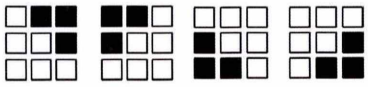


Fig. 3.5: パターンC

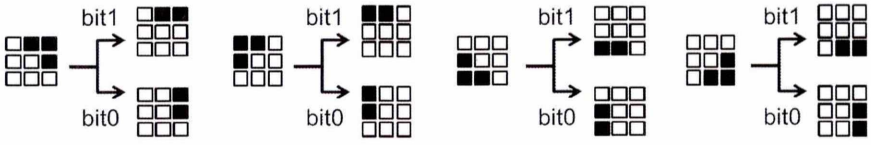


Fig. 3.6: パターンCの変更ルール

- パターンD

9 画素の内、L 字型の 4 ピクセルが白のパターンをパターン D とする。Fig.3.7 にエッジパターン 8 種を示し、Fig.3.8 に変更ルールを示す。

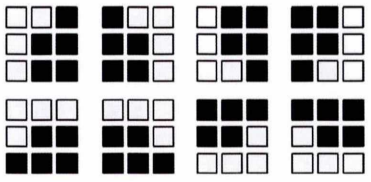


Fig. 3.7: パターンD

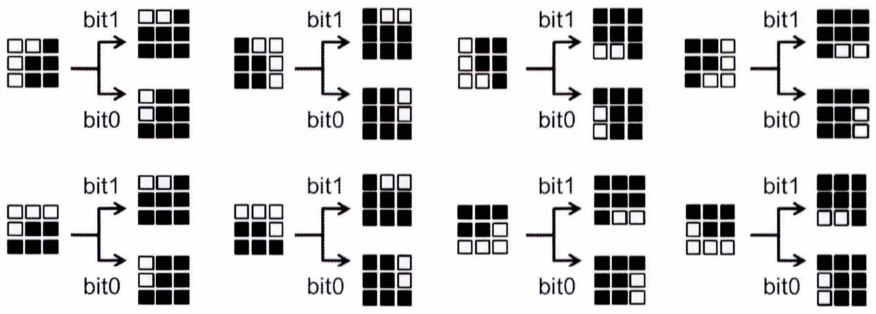


Fig. 3.8: パターンDの変更ルール

- パターンE

パターンD の8 パターンを白黒反転させたものをパターンE とする。Fig.3.9 にエッジパターン8種を示し、Fig.3.10 に変更ルールを示す。

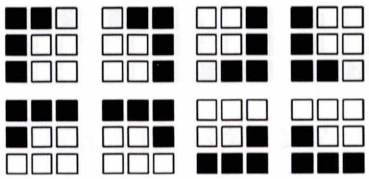


Fig. 3.9: パターンE

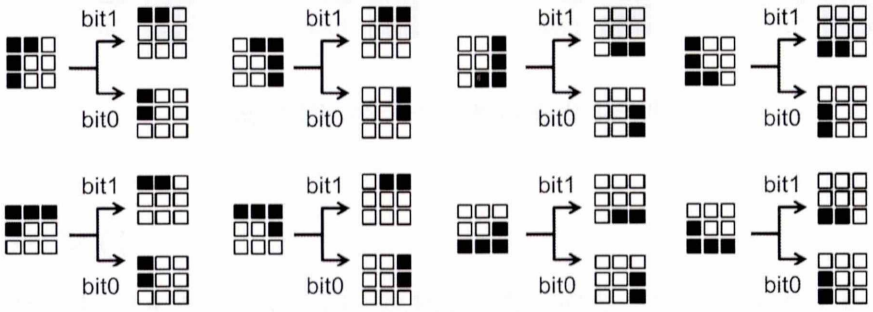


Fig. 3.10: パターンE の変更ルール

- パターンF

9 画素の内、鉤括弧型の 5 ピクセルが黒のパターンをパターンF とする。Fig.3.11 にエッジパターン 4 種を示し、Fig.3.12 に変更ルールを示す。

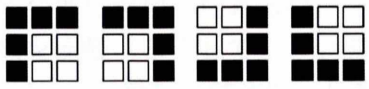


Fig. 3.11: パターンF

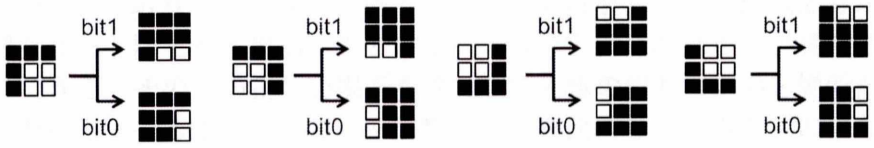


Fig. 3.12: パターンF の変更ルール

- パターンG

パターンF の 4 パターンを白黒反転させたものをパターンG とする。Fig.3.13 にエッジパターン 4 種を示し、Fig.3.14 に変更ルールを示す。

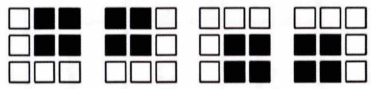


Fig. 3.13: パターンG

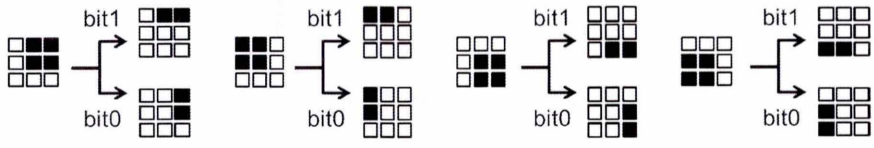


Fig. 3.14: パターンG の変更ルール

3.2　誤り訂正符号を用いた位置ずれ検出

スキャンした際の位置ずれに対応する方法として、埋め込みたい情報に誤り訂正符号を付加する。読み取り時には誤り訂正符号をデコードし本来の情報を得る。情報を埋め込む際はASCII コードから情報を 2 進数に変換し画像に埋め込む。そして読み取った時の誤り訂正符号によって得られる誤りビット数を比較することで正しい読み取り開始位置を検出する。今回は (16，8) 符号を使っている。（16，8）符号とは 2 ビットまでのランダム誤りと、バースト長が 3 ビットまでのバースト誤りを訂正することのできる誤り訂正符号のことである。半角英数字は 8bit で表され、さらに 8bit の冗長性を持たせることで 1 文字 16bit になり、2 倍の情報量を埋め込むことになる。検出するときの方法は、情報の読み取り開始位置を Fig.3.15 のように 9 個所から読み取りをおこなう。例えば、位置ずれをおこして斜線部分から読み取りを開始して誤りビット数を多く検出したとする。この場合斜線部は正しい読み取り開始位置でないと判断できる。そのため正しい場所を探ために斜線の周りの他の 8 箇所からも同じように読み取りを行う、そして 9 箇所のビット数の誤差を比較して一番誤りの少ない場所を探すという方法である。誤り訂正符号付加した画像 5 枚から読み取りを行い、全ての画像で”a” から読み取り開始した場合の誤りビット数が少ないか比較する。

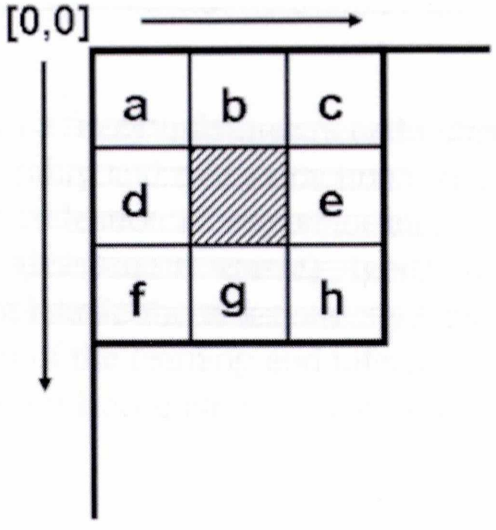


Fig. 3.15: 複数から読み取る例

3.3 ハフ変換を用いた傾き検出

前述の通り、エッジパターン法では 3×3 ピクセルごとにスキャンし、文字のエッジ部分のみに情報を埋め込む方法である。そのため、埋め込まれた情報は位置に依存するので、埋め込まれた情報を読み取るためスキャンする際に少しでも位置がずれてしまったり、傾いてしまうと正しい情報を取り出すことができないため、適切に文書の傾きを検出し、修正をしなければならない。一般的な A4 縦サイズの書類を 200dpi でスキャンした場合、画像サイズは 1654×2239 ピクセルとなる。エッジパターン法は、厳密には 1 ピクセルでもずれると読み取りができなくなるため、許容できる回転角度の誤差は、$\tan^{-1}(1/1654) = 0.035$ 度となる。このため少なくともこれ以上の精度で傾きを検出する必要があるが、できるだけ情報量の埋め込みの秘匿性を保持しなければいけない見地から次の手法を試みた。ハフ変換を用いて画像中の行から直線を検出して傾きを検出する手法である。情報埋め込みの秘匿性を損なうことなく傾きを検出するため、画像中の行から直線を複数検出しそれらをもとにタンジェントの値を計算することで傾きを計算した。Fig.3.16 にハフ変換を用いて行から直線を検出した様子を示す。

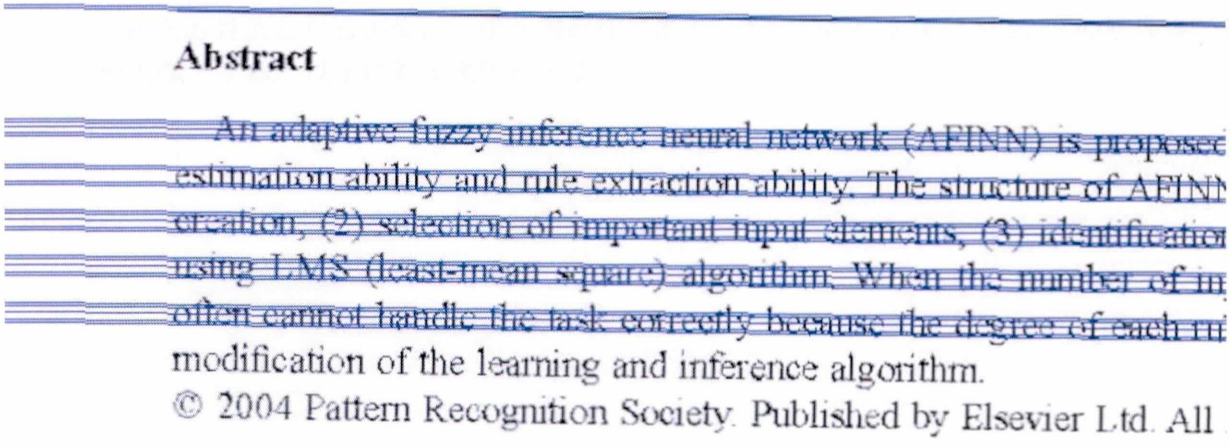


Fig. 3.16: ハフ変換を用いて直線を検出した様子

3.4 情報の読み取り

前述の手法により得られた傾きを用いて、bi-linear 法を用いて回転、拡大、縮小処理を行う事で情報埋め込み時の状態に近づける。しかし bi-linear 法を使用する際に二つの問題点があげられる。一つは文書画像に使用すると処理後 2 値画像ではなくなってしまうこと。二つ目は傾きを補正した後の画像が原画像と一致しないことである。一つ目の問題は 128 で閾値処理することにより再び 2 値化することが可能だが、二つ目の問題は傾きが大きくなるほど原画像との差が出る。差が出るということは情報が失われてしまい、正しく情報を読み取れなくなってしまう。傾きが少ない時では情報が失われていない部分があることで埋め込む情報を長い情報ではなく、URL 程度の短い情報にして連続で埋め込むことで情報を読み取ることも可能だと考えられる。しかし、傾きが増した時は別の方法を考える必要がある。そこで、補間画像と原画像を比較し画像間での誤差の出方、情報の壊れ方に規則性がないか調べた。

第4章　結果

この章では新たに追加したエッジパターンを組み合わせた時の埋め込み可能な情報量と画像劣化の比較、埋め込み可能な情報量の増加の比較、最適なエッジパターンの組み合わせ、誤り訂正符号による誤りビット数の比較、スキャン時の角度を求めるためにハフ変換を用いて画像中から直線を検出することで求められるタンジェントとそれに対応する角度の比較、画像補正後のエッジパターンの壊れ方の規則性とその結果を示す。

4.1　対象画像

本研究の対象となる情報を埋め込むために使用した5枚の画像をFig.4.1〜Fig.4.5に示す。ただし、研究の対象となる画像は2値文書画像であり、A4サイズの 1654×2239 の解像度200dpiの画像とする。以下に対象画像を示す。

PERGAMON

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition 37 (2004) 2049–2057

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Adaptive fuzzy inference neural network

Hitoshi Iyatomi*, Masafumi Hagiwara

Department of Information and Computer Science, Keio University, 3-14-1 Hiyoshi, Yokohama 223-8522, Japan

Received 6 February 2003; received in revised form 5 April 2004; accepted 5 April 2004

Abstract

An adaptive fuzzy inference neural network (AFINN) is proposed in this paper. It has self-construction ability, parameter estimation ability and rule extraction ability. The structure of AFINN is formed by the following four phases: (1) initial rule creation, (2) selection of important input elements, (3) identification of the network structure and (4) parameter estimation using LMS (least-mean square) algorithm. When the number of input dimension is large, the conventional fuzzy systems often cannot handle the task correctly because the degree of each rule becomes too small. AFINN solves such a problem by modification of the learning and inference algorithm.

Keywords: Neural network; Fuzzy inference; Machine learning; Fuzzy modeling and rule extraction

1. Introduction

Numeric analysis approach of fuzzy system was first presented by Takagi and Sugeno [1] and then a lot of studies have been made [1–11]. Since the systems using fuzzy theory can express rules or knowledge as "if-then" form, they have advantages such as they do not need mathematical analysis for modeling. However, they need the appropriate model construction and parameter selection [1–7]. This kind of fuzzy modeling problem is a troublesome work in general.

On the other hand, studies of fuzzy neural networks that combine both advantages of the fuzzy systems and the learning ability of the neural networks have been carried out. These techniques can alleviate the matter of fuzzy modeling by learning ability of neural networks and have been reported since around the beginning of 1990s [2,3]. Fuzzy neural networks can be applied not only for simple pattern classification but also meaningful fuzzy if-then rules creation; therefore, they can be put into practice for various applications. In the early stage of fuzzy neural network researches, Lin et al. [2] proposed one of the current prima models which decide the initial fuzzy model by Kohonen's self-organizing algorithm [8] and carry out parameter adjustment by back propagation algorithm. Also as a representative example, Jang et al. proposed ANFIS [9] in 1993. ANFIS applies a neural network in determination of the shape of membership functions and rule extraction. However, since it needs to divide the input data space in advance, accuracy of the system depends much on the achievement of this pre-processing. Wang et al. [10] reported an approach to acquire fuzzy rules by dividing input space. These techniques, however, do not consider the output data space, so the obtained rules should not be always reasonable. Nishina et al. proposed fuzzy inference neural network (FINN) [11]. FINN is a simple and effective fuzzy neural network that divides input–output data space and extracts fuzzy if-then rules automatically. Likewise many other fuzzy neural networks, FINN decides initial rules by self-organizing learning at first, then carries out merging rules based on Euclidean distance and uses LMS (least-mean square) algorithm

* Corresponding author. Tel.: +81-45-566-1762; fax: +81-45-566-1747.
E-mail address: iyatomi@soft.ics.keio.ac.jp (H. Iyatomi)

0031-3203/$ - see front matter © 2004 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2004.04.003

Fig. 4.1: 対象画像 1

法政大学大学院理工学・工学研究科紀要　Vol.55(2014 年 3 月)　　法政大学

改良エッジパターン法による文書画像への電子透かし

Digital Watermarking for Bi-level Image with Modefied Edge Pattern Method

三浦周平
shuhei MIURA

指導教員 鎗居仁

法政大学大学院工学研究科情報電子工学専攻修士課程

In this study, we proposed a new watermarking method for documents saved as bi-level images based on the edge-pattern method. The edge-pattern method is capable of putting hidden information in the document as bi-level images with little perceivable deterioration of the images. This method, however, has a limitation in the amount of embedded information and vulnerability for geometrical differences during the scanning the document. Our method improved the capable information around 3 times larger than the original and addressed the issues such as the shift and tilt during the scan with the (16, 8) error correcting code.
Key Words *: Bi-level Image, Text Image, Digital Water Marking, Edge Pattern*

1. まえがき

電子透かしとは文章や画像に，人の目には認識できないような微小な変化を施すことで情報を埋め込む技術であり，主に著作権保護などセキュリティ面で使用される．本研究は 2 値文章画像に対する独自の電子透かしを開発し，紙とデジタルの透過性の向上を目的とする．文章と同じ内容を画像に埋め込むことができれば高精度 OCR の実現が期待でき，web の URL や著作権情報を印刷した文章に埋め込むことで，それぞれ印刷した紙から web にアクセスし，関連データの取得や著作権の主張といった応用が期待できる．本研究では文章画像に対する電子透かしの手法として，エッジパターン法 [1] を使用する．この手法は文字のエッジ部分を微小に変化することで情報を埋め込む技術で，文章画像に対する従来の手法の中でも画像の劣化を抑えたまま情報量を多く埋め込むことができる．しかし，目的の情報量には不十分であることやスキャン時の紙の傾きや位置ずれによって正しい情報の読み取りができない欠点がある．そこで本研究では，エッジパターン法を改良した新たな手法を提案する．

2. エッジパターン法

エッジパターン法は，文章を 2 値画像とみなし情報埋め込み時に 3×3 のウィンドウを画像中に歪なりなくスキャンしていく．文字のエッジから検出したウィンドウパターンが予め定義された 12 個の内のパターンと一致した場合それをあるルールに従って変更することで情報を埋め込んでいく．Fig.1 に既定のパターンを示す．文章画像中をスキャンした際にこれらに該当するのパターンを bit1 や bit0 に該当するパターンに変化させることで情報を埋め込む．

3. 改良エッジパターン法

エッジパターン法は，文字のエッジを微小に変更し情報を埋め込むため画像劣化を抑えながら，従来の方法 [2][3] より多くの情報を埋め込める．しかし画像をスキャンする際の位置ずれや傾き，拡大縮小，歪みに対してとても弱い．そこで本研究では，(1) 秘匿性を保ったままに埋め込める情報量を増加させる．また (2) 誤り訂正符号を用いた位置ずれ補正とハフ変換を用いた傾き検出によってスキャン時の拡大縮小，回転による位置ずれの解決を試みた．

(1) パターンの増加による埋め込み情報量の増加

埋め込める情報量を増やすために，定めるウィンドウパターンを新たに 44 種を追加し，情報量に重点を置いた「情報量重視パターン（56 種）」を定義した．この中から画像の劣化と埋め込み情報量のバランスを考慮した「推奨パターン（28 種）」の定義も行った．A4 サイズの英字論文を 200dpi でスキャンし 2 値化した画像の一部分を拡大したものを Fig.2(a) とし，以下 (b) 既存パターン (c) 推奨パターン (d) 情報量重視パターン の結果を示す．(d) は，見た目の劣化がある程度感知できるが，(c) では劣化が気にならないレベルである．全 9 ページの英字論文を 200dpi でスキャンして画像化し，情報の埋め込み試験を行った埋め込み可能情報量の結果は Table.1 のようになった．推奨パターンを埋め込んだ場合，既存手法比平均で約 2.8 倍，情報量重視パターンを埋め込んだ場合，既存パターンに対し平均で約 12 倍となり，他の原稿を対象にした場合でも同様の傾向が見られた．

4. 誤り訂正符号を用いた位置ずれ補正

エッジパターン法は，3×3 のウィンドウパターンを元に情報の読み書きを行うためノイズや位置ずれに極めて弱いことから、正確な位置が読み込めない恐れがある。そこで，より冗長度の高い誤り訂正符号である (16,8) 符号を用い，情報の耐ノイズ性を高めると共に，位置ずれの検出にも利用した．(16,8) 符号は，8 ビットの情報に 8 ビットの訂正符号を付加する符号で，ちょうど英字 1 文字の情報毎に訂正符号を付加することを意味し，文字情報の埋め込みとの親和性も高い．この符号は，16bit 中の 2bit までのランダム誤りおよび 3bit のバースト誤りの訂正ができる．読み取り開始の位置推定には，読み取り開始位置候補の画素の近傍 3 × 3 の各画素を始点として復調を試み，誤りビット数の一番少ない位置を

Fig. 4.2: 対象画像 2

ネットワークアプリケーション設計論レポート(2010/4/21)

応用情報工学科
08x3088
三浦周平

1．はじめに

私の取り上げるグローバルにビジネスを展開する上で必要となる技術は、ネットワーク技術である。ネットワークで接続された環境では様々なデータ処理がおこなわれており、その処理をおこなうには高度な技術が必要となる。さらに大規模なデータを処理することによってユーザーが快適なネットワーク環境ですごせるのだ。そのため必要となる技術が備わっていないと今以上の発展はないと考えた。

2．技術概要

MapReduse とは、多数のマシンで効率的にデータ処理を行う目的で Google によって考案されたフレームワーク（一般的な機能をもった共通のコードを我々ユーザーが上書きしたり強化して、特定の機能をもたせようとする抽象概念のこと）である．開発者は分散処理の難しい部分を MapReduse に任せることで，少ない労力で大規模な処理を実行できる。

3．適用事例

大規模分散処理をおこなっている。そのため Google Earth でたくさんのリアルの写真を見ながら地図を確認し目的地を検索できるようになっている。さらにキーワード検索でも瞬時にたくさんの情報を処理しできる。

4．まとめ

MapReduseはたくさんのデータをより効率的に，安価に，簡単に処理できるようにするために編み出された．この機能により私たちの生活はよりかいてきになり楽しくネットワークを利用できる。たとえば、

・Google Earthのようなアプリケーションをさらに充実させ、開発することによる生活の補助
・ハードウェアを大量に購入することもなくネット上に保管し管理することのできるほどの容量を持たせ、セキュリティの充実。ラックに収めてネットワークにつないで管理するといった作業も必要もない．
・リソースが足りなくなることや，組織内のほかのメンバーとの共有について心配することもない．監視もチューニングも不要
・システムやアプリケーション・ソフトウェアのアップグレードに時間をかける必要もない．
・時間単位課金で実行できる．

Fig. 4.3: 対象画像 3

to adjust the parameters. Since the architecture and behavior of FINN are very applicable, it has been adopted as a basic component for image recognition and interpretation researches [12,13]. However, its fuzzy modeling for the target task is not always sufficient.

A lot of systems which aim at excellent fuzzy modeling and carry out input selection, rule creation and parameter estimation have been proposed [4–7,14,15]. Elimination of unnecessary rules and selection of efficient input elements can contribute the performance improvement, reduction of calculation cost and analysis of the obtained rules that is one of the most important merits of fuzzy systems. In these researches, Linkens et al. [6,7] reported effective input selection and rule creation method and showed the excellent results on their experiments. However, since these methods use fuzzy c-means (FCM) based algorithm as well as [16,17], they have to know the number of proper rules in advance. In addition, since the algorithms are complicated and their algorithms and results are presented only for single output system, they have difficulty if they are employed for many applications.

Generally, many fuzzy neural network models have common problems derived from their fundamental algorithm. For example, the systems which use gradient decent method sometimes reduce the width of the membership function to negative during the learning. The systems which employ basic fuzzy inference theory make the degree of each rule extremely small and often make it underflow when the dimension of the task is large. In such a situation, the learning and inference cannot be carried out correctly.

In this paper, we propose a new adaptive fuzzy inference neural network (AFINN) that alleviates these shortcomings of the conventional models. AFINN carries out appropriate model construction and solves fuzzy-neuro specific problems by modification of the fuzzy inference and learning algorithm.

Following four steps perform model construction of the AFINN:

1. determination of the initial rules by adaptive self-organizing learning;
2. selection of important input elements;
3. determination of the fuzzy neural network model by adaptive self-organizing learning;
4. parameter estimation using LMS learning.

In Section 2, structure and behavior of the proposed AFINN are described. In Section 3, the model construction of AFINN including input selection, rule creation and parameter estimation are explained. In Section 4, three examples are used to examine the efficiency of AFINN. Conclusions are summarized in the last section.

2. AFINN

To construct appropriate fuzzy model, the following problems should be solved:

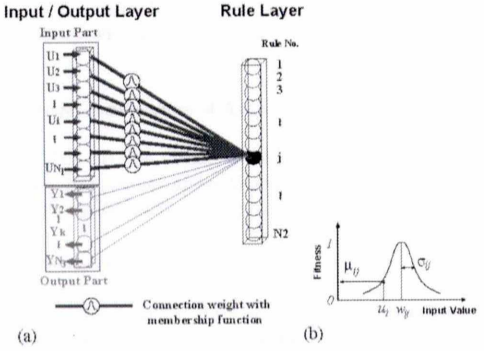


Fig. 1. (a) Structure of the AFINN and (b) its membership function.

1. identification of the optimum number of rules;
2. identification of the optimum element of inputs;
3. identification of the system parameter.

The proposed AFINN aims for the realization of above features and performs fuzzy-modeling based on modeling parameters such as ξ_{self} and ε_{self}, those are defined later. The modeling scheme is designed with the simple and versatile FINN architecture [11]. First, we explain the structure and the behavior of AFINN.

2.1. Structure of AFINN

An AFINN can divide input–output data space and provide appropriate rules automatically. Fig. 1 shows the structure of AFINN. It consists of two layers. One is the input–output (I/O) layer and another is the rule-layer. The I/O layer consists of the input-part and the output-part. Each node in the rule-layer represents one fuzzy rule. Weights from the input-part to the rule-layer and those from the rule-layer to the output-part are fully connected and they store fuzzy if-then rules. Membership functions as premise part are expressed in the weights. Each weight from the rule-layer to the output-part corresponds to the estimated value of each rule. In short, the weights from the input-part to the rule-layer indicate if-parts of fuzzy if-then rules and those from the rule-layer to the output-part indicate then-parts. The shapes of membership functions are adjusted automatically in the learning phase.

2.2. Behavior of AFINN

Suppose that the number of neurons in the input-part, which is equal to the dimension of the input data, is N_1, the number of rules is N_2, and the number of neurons in the output-part, which is equal to the dimension of the output

Fig. 4.4: 対象画像 4

data, is N_3. The input data to the AFINN is expressed as follows:

$$U = (u_1, u_2, \ldots, u_i, \ldots, u_{N_1}). \tag{1}$$

The subscripts i, j, and k refer to the nodes in the input-part, those in the rule-layer, and those in the output-part, respectively. Fig. 1(b) shows an example of a membership function. The bell-shaped membership function represents the if-part of fuzzy rule, which is placed between the ith input node and the jth node in the rule-layer. The membership function is expressed as

$$\mu_{ij} = \exp\left(-\frac{(u_i - w_{ij})^2}{\sigma_{ij}^2}\right),$$
$$i = (1, 2, \ldots, N_1), \quad j = (1, 2, \ldots, N_2), \tag{2}$$

where μ_{ij} is the membership value, w_{ij} is the center value of the membership function and σ_{ij} indicates the width adjusted in the parameter estimation phase explained in Section 3.4.

In the rule-layer, many conventional fuzzy systems calculate the degree of the rule by selecting minimum membership value or multiplying them as follows:

$$\rho_j = \min_i \mu_{ij}, \tag{3}$$

$$\rho_j = \prod_i^{N_1} \mu_{ij}. \tag{4}$$

These calculations, however, often tend to make ρ extremely small and sometimes they cause underflow when the dimension of the task is large. In such a situation, the learning and inference cannot be proceeded correctly. In order to solve such a problem, AFINN calculates the degree of the jth rule ρ_j as

$$\rho_j = \prod_i^{N_1} \mu_{ij}^{1/N_{adj}}. \tag{5}$$

Here, N_{adj} is the compensated factor relating to the input dimension. We discuss the efficiency of N_{adj} for high-dimensional data in Section 4.2.

Then, the inference result of the kth node in the output-part is calculated by the following equation:

$$\hat{y}_k = \frac{\sum_j^{N_2} (w_{jk}\rho_j)}{\sum_j^{N_2} \rho_j}, \quad k = (1, 2, \ldots, N_3), \tag{6}$$

where w_{jk} is the weight between the jth node in the rule-layer and the kth node in the output-part. The w_{jk} corresponds to the estimated value of the jth rule for the kth node in the output-part. The logical form of the fuzzy inference if-then rules is given such as

If u_1 is $\tilde{w}_{1j}$, and $\ldots, u_i$ is $\tilde{w}_{ij}, \ldots, u_{N_1}$ is $\tilde{w}_{N_1 j}$
then y_k is w_{jk},

where $\tilde{w}_{ij}$ means the value near w_{ij}. It should be noted here that it depends on the value of σ_{ij}.

3. Model construction of AFINN

AFINN has four phases to achieve appropriate input selection, rule creation and parameter adjustment. In this section we explain each of them.

3.1. Initial rule creation

The initial temporal rules are formed in this phase by adaptive self-organizing learning algorithm. This algorithm is developed based on that of Linkens [6] and is modified to the AFINN structure. It can construct variable structure in which the number of rules can be changed dynamically in response to incoming training data. The algorithm is expressed as follows:

Preparation. The lth input vector to the system $\boldsymbol{I}^l$ is defined as follows:

$$\boldsymbol{I}^l = \begin{bmatrix} U \\ Y \end{bmatrix}. \tag{7}$$

Here, the vector Y is the desired data vector in the output-part of the I/O layer described as

$$Y = (y_1^l, y_2^l, \ldots, y_k^l, \ldots, y_{N_3}^l)^{\mathrm{T}}. \tag{8}$$

The suffix l of $\boldsymbol{I}$ indicates the consecutive number of the learning vector ($1 \leqslant l \leqslant L$): the learning vector $\boldsymbol{I}$ consists of L sets of $(N_1 + N_3)$ dimensional vectors. They are normalized in [0,1] and inputted to the system.

The jth network weight vector $\boldsymbol{W}$ is defined to concatenate the weight vector from the ith input-part in the I/O layer to the jth node in the rule-layer, $\boldsymbol{w}_{ij} = (w_{1j}, w_{2j}, \ldots, w_{ij}, \ldots, w_{N_1 j})^{\mathrm{T}}$, and that from the jth node in the rule-layer to the kth output-part in the I/O layer, $\boldsymbol{w}_{ij} = (w_{j1}, w_{j2}, \ldots, w_{jk}, \ldots, w_{jN_3})^{\mathrm{T}}$. Note that $\boldsymbol{W}$ is $(N_1 + N_3)$-dimensional vector as well as $\boldsymbol{I}^l$.

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_{ij} \\ \boldsymbol{w}_{ij} \end{bmatrix}. \tag{9}$$

The number of current rule N_r is set to 0, the number of updated iterations for the jth rule defined as M_j is set to 0 $(\forall j)$ and the consecutive number of the input vector l is set to 1.

Step 1: The first input vector $\boldsymbol{I}^1$ is used as the first rule $\boldsymbol{W}_1$.

$$\boldsymbol{W}_1 = \boldsymbol{I}^1. \tag{10}$$

N_r is set to 1 and $l = l + 1$.

Step 2: The winner rule j^* is selected from existing N_r rules where Euclidian distance between the lth input vector

Fig. 4.5: 対象画像 5

4.2　情報量の増加

既存のエッジパターンのほかに新たに加わったエッジパターンA、B、C、D、E、F、Gの計44パターンを含む56パターンを使って情報の埋め込みを行った。結果画像を次に示し、そして劣化がわかりやすいように拡大画像を後に示す。


PERGAMON

Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition 37 (2004) 2049–2057

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Adaptive fuzzy inference neural network

Hitoshi Iyatomi*, Masafumi Hagiwara

Department of Information and Computer Science, Keio University, 3-14-1 Hiyoshi, Yokohama 223-8522, Japan

Received 6 February 2003; received in revised form 5 April 2004; accepted 5 April 2004

Abstract

An adaptive fuzzy inference neural network (AFINN) is proposed in this paper. It has self-construction ability, parameter estimation ability and rule extraction ability. The structure of AFINN is formed by the following four phases: (1) initial rule creation, (2) selection of important input elements, (3) identification of the network structure and (4) parameter estimation using LMS (least-mean square) algorithm. When the number of input dimension is large, the conventional fuzzy systems often cannot handle the task correctly because the degree of each rule becomes too small. AFINN solves such a problem by modification of the learning and inference algorithm.

Keywords: Neural network; Fuzzy inference; Machine learning; Fuzzy modeling and rule extraction

1. Introduction

Numeric analysis approach of fuzzy system was first presented by Takagi and Sugeno [1] and then a lot of studies have been made [1–11]. Since the systems using fuzzy theory can express rules or knowledge as "if-then" form, they have advantages such as they do not need mathematical analysis for modeling. However, they need the appropriate model construction and parameter selection [1–7]. This kind of fuzzy modeling problem is a troublesome work in general.

On the other hand, studies of fuzzy neural networks that combine both advantages of the fuzzy systems and the learning ability of the neural networks have been carried out. These techniques can alleviate the matter of fuzzy modeling by learning ability of neural networks and have been reported since around the beginning of 1990s [2,3]. Fuzzy neural networks can be applied not only for simple pattern classification but also meaningful fuzzy if-then rules creation; therefore, they can be put into practice for various applications. In the early stage of fuzzy neural network researches, Lin et al. [2] proposed one of the current prima models which decide the initial fuzzy model by Kohonen's self-organizing algorithm [8] and carry out parameter adjustment by back propagation algorithm. Also as a representative example, Jang et al. proposed ANFIS [9] in 1993. ANFIS applies a neural network in determination of the shape of membership functions and rule extraction. However, since it needs to divide the input data space in advance, accuracy of the system depends much on the achievement of this pre-processing. Wang et al. [10] reported an approach to acquire fuzzy rules by dividing input space. These techniques, however, do not consider the output data space, so the obtained rules should not be always reasonable. Nishina et al. proposed fuzzy inference neural network (FINN) [11]. FINN is a simple and effective fuzzy neural network that divides input–output data space and extracts fuzzy if-then rules automatically. Likewise many other fuzzy neural networks, FINN decides initial rules by self-organizing learning at first, then carries out merging rules based on Euclidean distance and uses LMS (least-mean square) algorithm

* Corresponding author. Tel.: +81-45-566-1762; fax: +81-45-566-1747.
E-mail address: iyatomi@soft.ics.keio.ac.jp (H. Iyatomi)

0031-3203/$ - see front matter © 2004 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2004.04.003

Fig. 4.6: 対象画像 1 に全パターンを使って情報を埋め込んだ画像

法政大学大学院理工学・工学研究科紀要　Vol.55(2014年3月)　　法政大学

改良エッジパターン法による文書画像への電子透かし

Digital Watermarking for Bi-level Image with Modefied Edge Pattern Method

二浦周平
simhei MIURA

指導教員 細原七

法政大学大学院工学研究科情報電子工学専攻修士課程

In this study, we proposed a new watermarking method for documents saved as bi-level images based on the edge-pattern method. The edge-pattern method is capable of putting hidden information in the document as bi-level images with little perceivable deterioration of the images. This method, however, has a limitation in the amount of embedded information and vulnerability for geometrical differences during the scanning the document. Our method improved the capable information around 3 times larger than the original and addressed the issues such as the shift and tilt during the scan with the (16, 8) error correcting code.

***Key Words** : Bi-level Image, Text Image, Digital Water Marking, Edge Pattern*

1. まえがき

電子透かしとは文章や画像に，人の目には認識できないような微小な変化を施すことで情報を埋め込む技術であり，主に著作権保護などセキュリティ面で使用される．本研究は2値文書画像に対する独自の電子透かしを開発し，紙とデジタルの透過性の向上を目的とする．文章と見た目の変化を画像に埋め込むことができれば高精度OCRの実現が期待でき，webのURLや著作権情報を印刷した文章に埋め込むことで，それを紙印刷した紙からwebにアクセスし，関連データの取得や著作権の主張といった応用が期待できる．本研究では，文書画像に対する電子透かしの手法として，エッジパターン法[1]を使用する．この手法は文字のエッジ部分を微小に変化することで情報を埋め込む技術で，文書画像に対する従来の手法の中でも画像の劣化を抑えたまま情報量を多く埋め込むことができる．しかし，目的の情報量には十分であることやスキャン時の紙の斜きや位置ずれによって正しい情報の読み取りができない欠点がある．そこで本研究では，エッジパターン法を改良した新たな手法を提案する．

2. エッジパターン法

エッジパターン法は，文章を2値画像とみなし情報埋め込み時に3×3のウィンドウを画像中に重なりなくスキャンしていく．文字のエッジから検出したウィンドウパターンが予め定義された12個の内のパターンと一致した場合それをあるルールに従って変更することで情報を埋め込んでいく．Fig.1に既定のパターンを示す．文章画像中をスキャンした際にこれらに該当するのパターンをbit1やbit0に該当するパターンに変化させることで情報を埋め込む．

3. 改良エッジパターン法

エッジパターン法は，文字のエッジを微小に変更し情報を埋め込むため画像劣化を抑えながら，従来の方法[2][3]よりもその情報を埋め込める．しかし画像をスキャンする際の位置ずれや傾き，拡大縮小，歪みに対してとても弱い．そこで本研究では，(1)劣化を保ったままに埋め込める情報量を増加させる．また(2)誤り訂正符号を用いた位置ずれ補正とパス変換を用いた傾き検出によってスキャン時の拡大縮小，回転による位置ずれの解決を試みた．

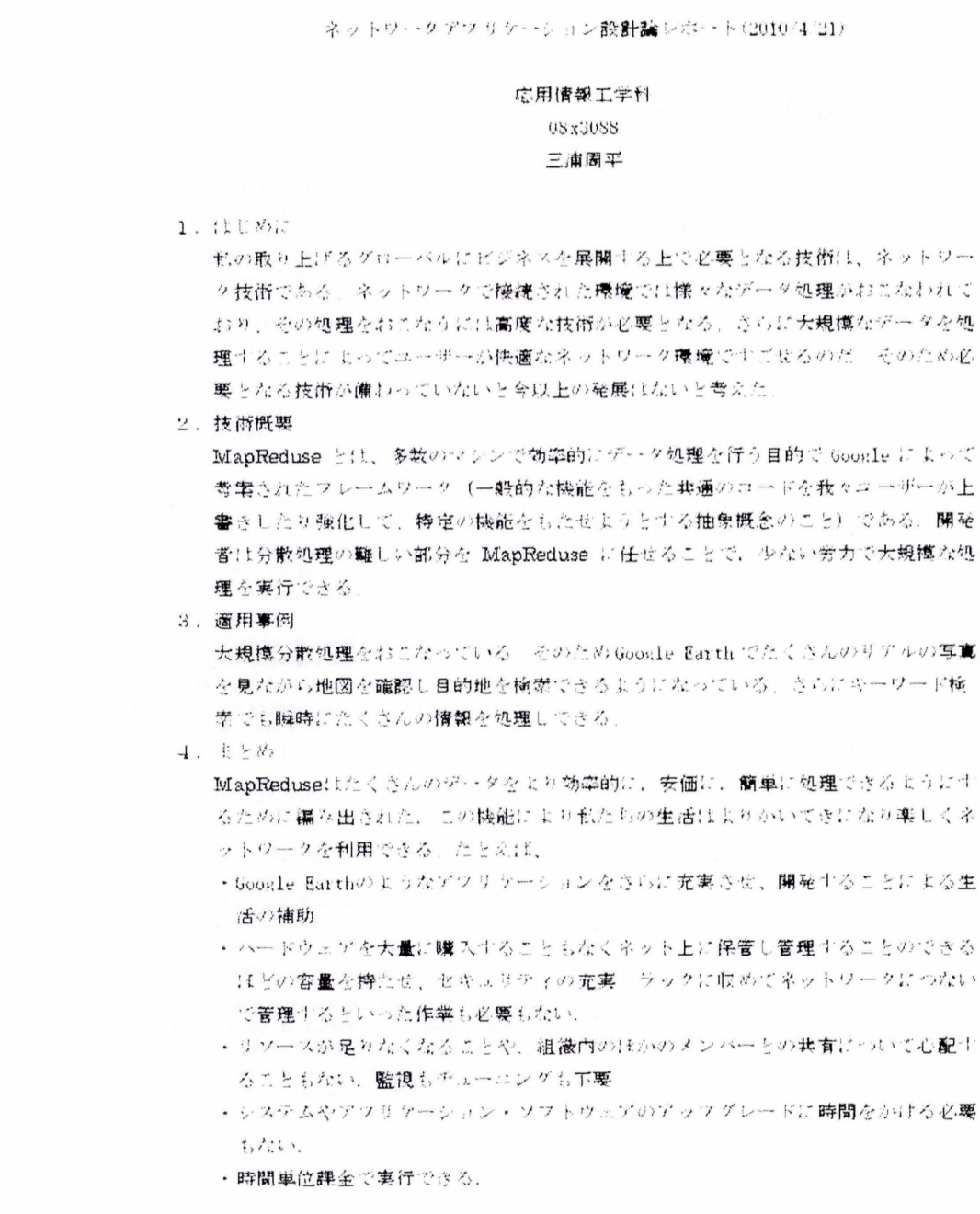
(1) パターンの増加による埋め込み情報量の増加

埋め込める情報量を増やすために，定めるウィンドウパターンを新たに11種を追加し，情報量に由来を置いた「情報量由来パターン(56種)」を定義した．この中から画像の劣化と埋め込み情報量のバランスを考慮した「推奨パターン(25種)」の定義も行った．A4サイズの英字論文を200dpiでスキャンし2値化した画像の一部分を拡大したものをFig.2(a)とし，以下(b)既存パターン(c)推奨パターン(d)情報量由来パターンの結果を示す．(d)は，見た目の劣化がある程度感知できるが，(c)では劣化が気にならないレベルである．予めページの英字論文を200dpiでスキャンして画像化し，情報の埋め込み試験を行った埋め込み可能情報量の結果はTable.1のようになった．推奨パターンを埋め込んだ場合，既存手法比平均で約2.8倍，情報量由来パターンを埋め込んだ場合，既存パターンに対し平均で約12倍となり，他の原稿を対象にした場合でも同様の結果が見られた．

4. 誤り訂正符号を用いた位置ずれ補正

エッジパターン法は，3×3のウィンドウパターンを元に情報の読み書きを行うためノイズや位置ずれに極めて弱いことから，僅かな位置ずれも読み込めない恐れがある．そこで，より冗長度の高い誤り訂正符号である(16,8)符号を用い，情報の耐ノイズ性を高めると共に，位置ずれの検出にも利用した．(16,8)符号は，8ビットの情報に8ビットの訂正符号を付加する符号で，ちょうど英字1文字の情報毎に訂正符号を付加することを意味し，文字情報の埋め込みとの親和性もよい．この符号は，16bit中の2bitまでのランダム誤りおよび3bitのバースト誤りの訂正ができる．読み取り開始位置探索横の画素の遷移3×3の注画素を始点として仮説を試み，誤りビット数の一番少ない位置を

Fig. 4.7: 対象画像2に全パターンを使って情報を埋め込んだ画像



ネットワークアプリケーション設計論レポート(2010/4/21)

応用情報工学科

08x308S

三浦周平

1．はじめに

私の取り上げるグローバルにビジネスを展開する上で必要となる技術は、ネットワーク技術である。ネットワークで接続された環境では様々なデータ処理がおこなわれており、その処理をおこなうには高度な技術が必要となる。さらに大規模なデータを処理することによってユーザーが快適なネットワーク環境で生活できるのだ。そのため必要となる技術が備わっていないと今以上の発展はないと考えた。

2．技術概要

MapReduse とは、多数のマシンで効率的にデータ処理を行う目的で Google によって考案されたフレームワーク（一般的な機能をもった共通のコードを我々ユーザーが上書きしたり強化して、特定の機能をもたせようとする抽象概念のこと）である。開発者は分散処理の難しい部分を MapReduse に任せることで、少ない労力で大規模な処理を実行できる。

3．適用事例

大規模分散処理をおこなっている。そのため Google Earth でたくさんのリアルの写真を見ながら地図を確認し目的地を検索できるようになっている。さらにキーワード検索でも瞬時にたくさんの情報を処理してできる。

4．まとめ

MapReduse はたくさんのデータをより効率的に，安価に，簡単に処理できるようにするために編み出された。この技能により私たちの生活はよりかいてきになり楽しくネットワークを利用できる。たとえば、

- Google Earthのようなアプリケーションをさらに充実させ、開発することによる生活の補助
- ハードウェアを大量に購入することもなくネット上に保管し管理することのできるほどの容量を持たせ、セキュリティの充実　ラックに収めてネットワークにつないで管理するといった作業も必要もない。
- リソースが足りなくなることや、組織内のほかのメンバーとの共有について心配することもない。監視もチューニングも不要
- システムやアプリケーション・ソフトウェアのアップグレードに時間をかける必要もない。
- 時間単位課金で実行できる。

Fig. 4.8: 対象画像 3 に全パターンを使って情報を埋め込んだ画像

to adjust the parameters. Since the architecture and behavior of FINN are very applicable, it has been adopted as a basic component for image recognition and interpretation researches [12,13]. However, its fuzzy modeling for the target task is not always sufficient.

A lot of systems which aim at excellent fuzzy modeling and carry out input selection, rule creation and parameter estimation have been proposed [4–7,14,15]. Elimination of unnecessary rules and selection of efficient input elements can contribute the performance improvement, reduction of calculation cost and analysis of the obtained rules that is one of the most important merits of fuzzy systems. In these researches, Linkens et al. [6,7] reported effective input selection and rule creation method and showed the excellent results on their experiments. However, since these methods use fuzzy c-means (FCM) based algorithm as well as [16,17], they have to know the number of proper rules in advance. In addition, since the algorithms are complicated and their algorithms and results are presented only for single output system, they have difficulty if they are employed for many applications.

Generally, many fuzzy neural network models have common problems derived from their fundamental algorithm. For example, the systems which use gradient decent method sometimes reduce the width of the membership function to negative during the learning. The systems which employ basic fuzzy inference theory make the degree of each rule extremely small and often make it underflow when the dimension of the task is large. In such a situation, the learning and inference cannot be carried out correctly.

In this paper, we propose a new adaptive fuzzy inference neural network (AFINN) that alleviates these shortcomings of the conventional models. AFINN carries out appropriate model construction and solves fuzzy-neuro specific problems by modification of the fuzzy inference and learning algorithm.

Following four steps perform model construction of the AFINN:

1. determination of the initial rules by adaptive self-organizing learning;
2. selection of important input elements;
3. determination of the fuzzy neural network model by adaptive self-organizing learning;
4. parameter estimation using LMS learning.

In Section 2, structure and behavior of the proposed AFINN are described. In Section 3, the model construction of AFINN including input selection, rule creation and parameter estimation are explained. In Section 4, three examples are used to examine the efficiency of AFINN. Conclusions are summarized in the last section.

2. AFINN

To construct appropriate fuzzy model, the following problems should be solved:

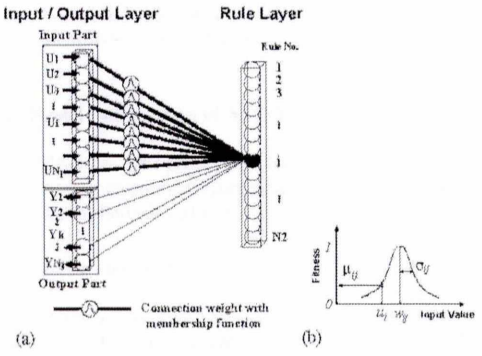


Fig. 1. (a) Structure of the AFINN and (b) its membership function.

1. identification of the optimum number of rules;
2. identification of the optimum element of inputs;
3. identification of the system parameter.

The proposed AFINN aims for the realization of above features and performs fuzzy-modeling based on modeling parameters such as ξ_{self} and ε_{self}, those are defined later. The modeling scheme is designed with the simple and versatile FINN architecture [11]. First, we explain the structure and the behavior of AFINN.

2.1. Structure of AFINN

An AFINN can divide input–output data space and provide appropriate rules automatically. Fig. 1 shows the structure of AFINN. It consists of two layers. One is the input–output (I/O) layer and another is the rule-layer. The I/O layer consists of the input-part and the output-part. Each node in the rule-layer represents one fuzzy rule. Weights from the input-part to the rule-layer and those from the rule-layer to the output-part are fully connected and they store fuzzy if-then rules. Membership functions as premise part are expressed in the weights. Each weight from the rule-layer to the output-part corresponds to the estimated value of each rule. In short, the weights from the input-part to the rule-layer indicate if-parts of fuzzy if-then rules and those from the rule-layer to the output-part indicate then-parts. The shapes of membership functions are adjusted automatically in the learning phase.

2.2. Behavior of AFINN

Suppose that the number of neurons in the input-part, which is equal to the dimension of the input data, is N_1, the number of rules is N_2, and the number of neurons in the output-part, which is equal to the dimension of the output

Fig. 4.9: 対象画像 4 に全パターンを使って情報を埋め込んだ画像

data, is N_3. The input data to the AFINN is expressed as follows:

$$U = (u_1, u_2, \ldots, u_i, \ldots, u_{N_1}). \tag{1}$$

The subscripts i, j, and k refer to the nodes in the input-part, those in the rule-layer, and those in the output-part, respectively. Fig. 1(b) shows an example of a membership function. The bell-shaped membership function represents the if-part of fuzzy rule, which is placed between the ith input node and the jth node in the rule-layer. The membership function is expressed as

$$\mu_{ij} = \exp\left(-\frac{(u_i - w_{ij})^2}{\sigma_{ij}^2}\right),$$
$$i = (1, 2, \ldots, N_1), \quad j = (1, 2, \ldots, N_2), \tag{2}$$

where μ_{ij} is the membership value, w_{ij} is the center value of the membership function and σ_{ij} indicates the width adjusted in the parameter estimation phase explained in Section 3.4.

In the rule-layer, many conventional fuzzy systems calculate the degree of the rule by selecting minimum membership value or multiplying them as follows:

$$\rho_j = \min_i \mu_{ij}, \tag{3}$$

$$\rho_j = \prod_i^{N_1} \mu_{ij}. \tag{4}$$

These calculations, however, often tend to make ρ extremely small and sometimes they cause underflow when the dimension of the task is large. In such a situation, the learning and inference cannot be proceeded correctly. In order to solve such a problem, AFINN calculates the degree of the jth rule ρ_j as

$$\rho_j = \prod_i^{N_1} \mu_{ij}^{1/N_{adj}}. \tag{5}$$

Here, N_{adj} is the compensated factor relating to the input dimension. We discuss the efficiency of N_{adj} for high-dimensional data in Section 4.2.

Then, the inference result of the kth node in the output-part is calculated by the following equation:

$$\hat{y}_k = \frac{\sum_j^{N_2} (w_{jk}\rho_j)}{\sum_j^{N_2} \rho_j}, \quad k = (1, 2, \ldots, N_3), \tag{6}$$

where w_{jk} is the weight between the jth node in the rule-layer and the kth node in the output-part. The w_{jk} corresponds to the estimated value of the jth rule for the kth node in the output-part. The logical form of the fuzzy inference if-then rules is given such as

If u_1 is $\tilde{w}_{1j}$, and $\ldots, u_i$ is $\tilde{w}_{ij}, \ldots, u_{N_1}$ is $\tilde{w}_{N_1 j}$
then y_k is w_{jk},

where $\tilde{w}_{ij}$ means the value near w_{ij}. It should be noted here that it depends on the value of σ_{ij}.

3. Model construction of AFINN

AFINN has four phases to achieve appropriate input selection, rule creation and parameter adjustment. In this section we explain each of them.

3.1. Initial rule creation

The initial temporal rules are formed in this phase by adaptive self-organizing learning algorithm. This algorithm is developed based on that of Linkens [6] and is modified to the AFINN structure. It can construct variable structure in which the number of rules can be changed dynamically in response to incoming training data. The algorithm is expressed as follows:

Preparation: The lth input vector to the system $\boldsymbol{I}^l$ is defined as follows:

$$\boldsymbol{I}^l = \begin{bmatrix} U \\ Y \end{bmatrix}. \tag{7}$$

Here, the vector Y is the desired data vector in the output-part of the I/O layer described as

$$Y = (y_1^l, y_2^l, \ldots, y_k^l, \ldots, y_{N_3}^l)^{\mathrm{T}}. \tag{8}$$

The suffix l of $\boldsymbol{I}$ indicates the consecutive number of the learning vector ($1 \leqslant l \leqslant L$): the learning vector $\boldsymbol{I}$ consists of L sets of $(N_1 + N_3)$ dimensional vectors. They are normalized in [0,1] and inputted to the system.

The jth network weight vector $\boldsymbol{W}$ is defined to concatenate the weight vector from the ith input-part in the I/O layer to the jth node in the rule-layer, $\boldsymbol{w}_{ij} = (w_{1j}, w_{2j}, \ldots, w_{ij}, \ldots, w_{N_1 j})^{\mathrm{T}}$, and that from the jth node in the rule-layer to the kth output-part in the I/O layer, $\boldsymbol{w}_{ij} = (w_{j1}, w_{j2}, \ldots, w_{jk}, \ldots, w_{jN_3})^{\mathrm{T}}$. Note that $\boldsymbol{W}$ is $(N_1 + N_3)$-dimensional vector as well as $\boldsymbol{I}^l$.

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_{ij} \\ \boldsymbol{w}_{ij} \end{bmatrix}. \tag{9}$$

The number of current rule N_r is set to 0, the number of updated iterations for the jth rule defined as M_j is set to 0 ($\forall j$) and the consecutive number of the input vector l is set to 1.

Step 1: The first input vector $\boldsymbol{I}^1$ is used as the first rule $\boldsymbol{W}_1$.

$$\boldsymbol{W}_1 = \boldsymbol{I}^1. \tag{10}$$

N_r is set to 1 and $l = l + 1$.

Step 2: The winner rule j^* is selected from existing N_r rules where Euclidian distance between the lth input vector

Fig. 4.10: 対象画像5に全パターンを使って情報を埋め込んだ画像

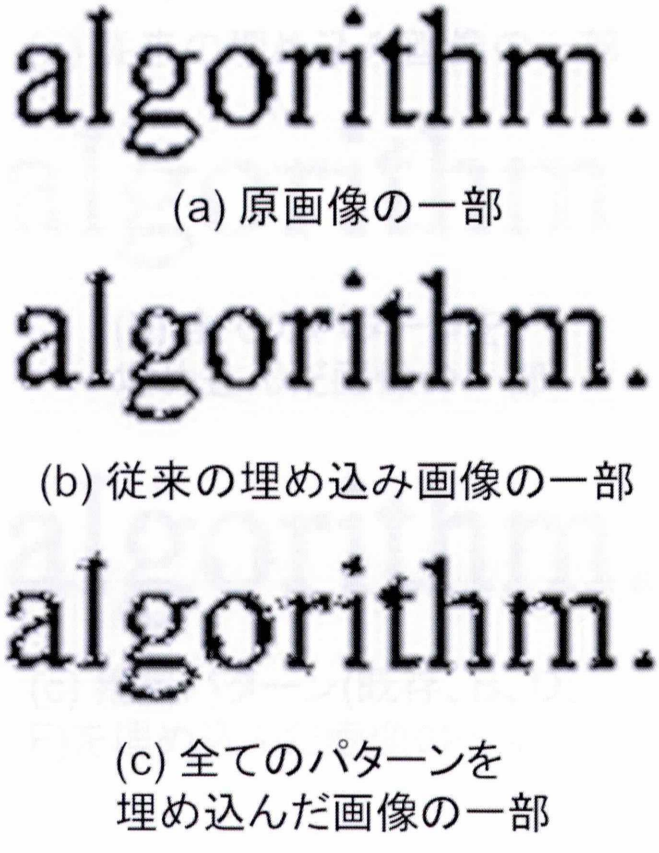


Fig. 4.11: Fig.4.6 を拡大したときの劣化の比較

全てのエッジパターンを使用して情報を埋め込んだ Fig.4.1 から Fig.4.5 の 5 枚の画像からわかるように等倍の画像だと劣化は目立たないが、Fig.4.11 のように拡大してみると全てのエッジパターンを使用した埋め込みでは劣化が見られる。そこで少しでも劣化を抑えるためにエッジパターンをいくつか組み合わせた推奨パターン (既存のパターン、パターン B、D、F) を使用して情報の埋め込みを行いその拡大画像をつかって比較を行う。その結果を Fig.4.12 に示す。

Fig. 4.12: 全パターンと推奨パターンとの劣化の比較

Table 4.1: 既存パターンと全パターンによる埋め込み可能な情報量の増加倍率

	既存パターン	全パターン	倍率
画像 1	13,179	51,458	3.90
画像 2	3,066	19,886	6.49
画像 3	12,403	46,460	3.75
画像 4	12,555	55,416	4.41
画像 5	11,640	54,341	4.67

Table 4.2: 既存パターンと推奨パターン (B,D,F) による埋め込み可能な情報量の増加倍率

	既存パターン	推奨パターン	倍率
画像 1	13,179	24,778	1.88
画像 2	3,066	7,252	2.37
画像 3	12,403	21,698	1.75
画像 4	12,555	23,880	1.90
画像 5	11,640	22,861	1.96

ここでは、既存パターンと全パターン、推奨パターンの情報量の増加倍率を比較していく。すべてのパターンを埋め込むと、拡大した画像では劣化が大きいことが確認できる。しかし、等倍の画像を見てみると、ほとんどと言っていいほど劣化が目立たない。Table 4.1 から既存パターンと埋め込める情報量の差を比べてみると、平均して 4.6 倍の情報量を埋め込むことが可能になった。これはビット数で表すと 35,000 ビット近く平均して増えており、従来のものより文字にして 4,400 文字近く埋め込めることができる。このことから埋め込みたい情報量が多い場合はすべてのパターンを使用できると考えられる。

画像の劣化を抑えつつ情報を埋め込むために、エッジパターン中の黒が多いパターンを組合せた推奨パターン (既存のパターン、パターン B、D、F) を使用したものとも比較した。その結果が Table 4.2 である。情報量に関して、すべてのパターンを使用したものに比べ約半分になっているが、文字数にして約 1,670 文字近く埋め込める。推奨パターンを使用すると情報量が減ってしまう半面、拡大画像ではほとんど劣化を抑え従来のものより約 2 倍の情報量を埋め込めることがわかった。

4.3 誤り訂正符号を用いた位置ずれの検出

位置ずれの検出を行うために埋め込みたい情報に誤り訂正符号を付加することで、位置ずれに対応した。誤り訂正符号付きの情報を埋め込んだ画像を読み取る際にFig.3.15のように9か所から読み取りを行う、斜線部が誤って読み取りを開始した位置として、その周りの8か所からも読み取りを行うことで誤りビット数を比較し正しい読み取り開始位置を検出できるか調べた。結果をTable 4.3に示す。表からわかるように9か所から読み取りを開始した結果”a”だけ誤りビット数が検出されていないことから”a”が正しい読み取り開始位置だとわかった。

Table 4.3: 位置ずれによる誤りビット数の比較

	a	b	c	d	斜線	e	f	g	h
画像1	0	180	127	193	145	171	166	169	144
画像2	0	133	157	155	167	116	192	164	129
画像3	0	168	141	177	158	200	231	131	143
画像4	0	173	120	176	143	125	186	144	134
画像5	0	153	143	180	165	179	185	160	179

4.4　ハフ変換を用いた傾き検出

スキャン時の傾きに対応するために、情報埋め込みの秘匿性を損なうことなく傾きを検出するためにハフ変換を用いて画像中の行から直線を複数検出しそれらをもとに傾きを求めた。Table 4.4 は 100 枚の 2 値画像を 0.1〜5 度傾け傾きを検出した時の平均誤差、標準偏差、最大誤差、誤り傾き数である。画像から情報を正常に読み取る時に、1 ピクセルずれてしまうと読み取れなくなってしまうのでこの範囲にズレを収めたい。1 ピクセル分の傾きを許容誤差角度として求めると $\tan^{-1}(1/1654) = 0.035$ 度となり検出した傾きをこの範囲以内で収めたい。Table 4.4 から 0.5〜5.0 度まではほとんど正しく傾きを求めることができたが、0.1 度のときに傾きを正確に求めることができていない。

Table 4.4: 画像（100 枚）を回転させハフ変換を用いて傾きを検出したときの平均誤差, 標準偏差, 最大誤差, 誤り傾き数の内訳

傾き	平均誤差	標準偏差	最大誤差	誤り傾き数
0.1	0.236	0.377	0.903	83
0.5	0.508	0.179	1.005	3
1.0	0.028	0.109	0.505	0
1.5	0.044	0.140	0.506	4
2.0	0.027	0.098	0.495	3
5.0	0.077	0.164	0.513	1

4.5 情報の読み取り

前述の手法により得られた傾きから、bi-linear 法を用いて補正を行う事で、情報埋め込み時の状態に近づける。しかし bi-linear 法を使用すると傾きを補正した後の画像が原画像と一致せず、(16,8) 符号を使用しても正しい情報を読み取れなくなる。補正前に埋め込まれていた情報が補正により失われることや出現することで 16bit の情報の順番が変わってしまうためである。Table 4.5 に縦を入力側、横を出力側とした補間後の埋め込み情報の対応表を示す。入力側で bit「1」であったところが出力側で bit「0」に変化することや bit「0」が「1」に変化することはないが、bit 1 や 0 が情報を持たないエッジパターンに変わっている。これは、埋め込まれた誤り符号付き情報 16bit の情報の一部が失われることになり 16bit の並びが変わってしまう。次に情報を持たないエッジパターンが bit「1」「0」を表すエッジパターンに変わると 16bit 中に存在しなかったデータが出てきてしまうため正しく情報を読み取れなくなってしまうことがわかった。

Table 4.5: エッジパターン別誤認識表（bit）

入力＼出力	[1]	[0]	white	corner	[-]
[1]	2403.4	0	17.2	0.8	526.1
[0]	0	2946.8	27.3	16.3	181.2
white	0	0	299879.1	0	0
corner	0	0	0	149	3.6
[-]	489.6	246.8	1485.5	25.5	36337.1
確率	0.8	0.9	0.9	0.7	0.9

そこで試験段階ではあるが、情報埋め込み画像（入力画像）と補間画像とを比較し画像間での誤差の出方に規則性がないか検証した。それは、bit 1 と 0 を表すエッジパターンに注目し情報埋め込み直後の画像を入力画像、補間後の画像を出力画像とした時入力側と出力側の同じ座標にあるエッジパターンを比較し誤っているかいないか、誤っていた場合誤った後のパターンが何になりそれがどのくらいあるのか調べた。その結果を Fig.4.13〜Fig.4.28 と Table 4.6〜Table 4.21 に示す。それぞれの図は、入力側のエッジパターンが補正後に変化したエッジパターンの遷移図を表しさらに各エッジパターンをわかりやすくするために数値化した。その値を用いて表にしたのが Table 4.6〜Table 4.21 である。

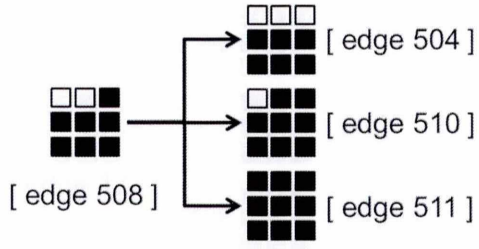


Fig. 4.13: bit「1」を表すエッジパターン（edge508）の補間後の遷移図

Table 4.6: bit「1」を表すエッジパターン（edge508）の補間後の遷移数

出力 / 入力	504	508	510	511	合計
508	46	867	215	1	1129
その他	0	4	0	0	4

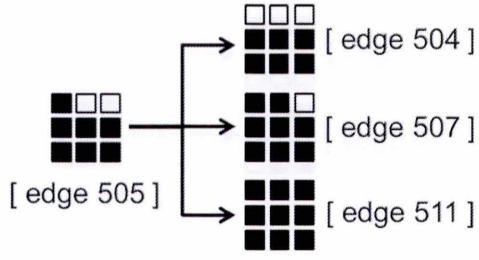


Fig. 4.14: bit「1」を表すエッジパターン（edge505）の補間後の遷移図

Table 4.7: bit「1」を表すエッジパターン（edge505）の補間後の遷移数

出力 / 入力	504	505	507	511	合計
505	22	1365	217	5	1609
その他	0	4	0	0	4

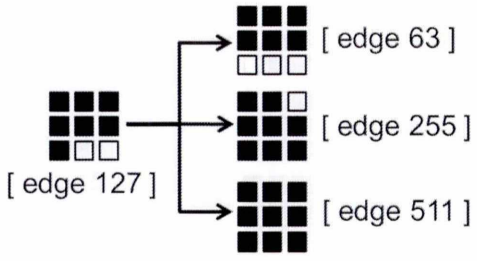


Fig. 4.15: bit「1」を表すエッジパターン（edge 127 ）の補間後の遷移図

Table 4.8: bit「1」を表すエッジパターン（edge 127 ）の補間後の遷移数

出力 / 入力	63	127	255	511	合計
127	15	620	136	1	772
その他	0	3	0	0	3

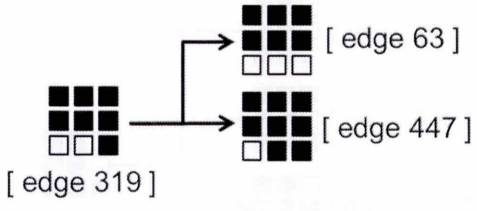


Fig. 4.16: bit「1」を表すエッジパターン（edge 319 ）の補間後の遷移図

Table 4.9: bit「1」を表すエッジパターン（edge 319 ）の補間後の遷移数

出力 / 入力	63	319	447	合計
319	31	1278	206	1515
その他	0	1	0	1

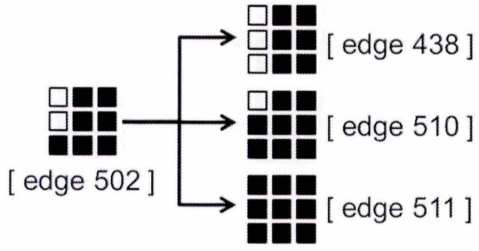


Fig. 4.17: bit「0」を表すエッジパターン（edge 502 ）の補間後の遷移図

Table 4.10: bit「0」を表すエッジパターン（edge 502 ）の補間後の遷移数

入力＼出力	438	502	510	511	合計
502	58	977	93	1	1129
その他	0	15	0	0	15

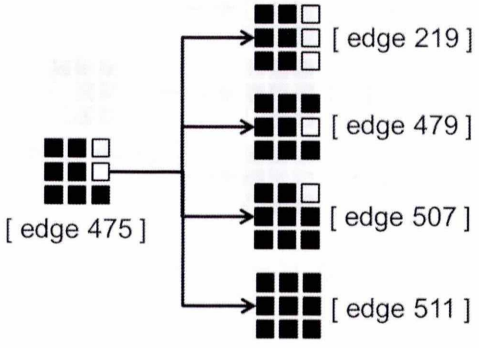


Fig. 4.18: bit「0」を表すエッジパターン（edge 475 ）の補間後の遷移図

Table 4.11: bit「0」を表すエッジパターン（edge 475 ）の補間後の遷移数

入力＼出力	219	475	479	507	511	合計
475	75	1419	2	109	4	1609
その他	0	17	0	0	0	17

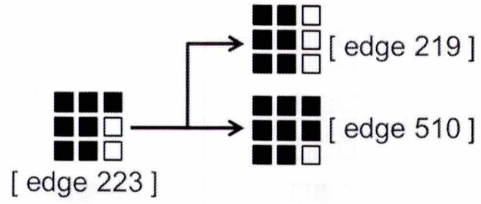


Fig. 4.19: bit「0」を表すエッジパターン（edge 223 ）の補間後の遷移図

Table 4.12: bit「0」を表すエッジパターン（edge 223 ）の補間後の遷移数

入力＼出力	219	223	255	合計
223	22	687	63	772
その他	0	11	0	11

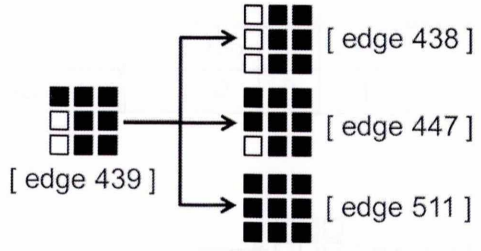


Fig. 4.20: bit「0」を表すエッジパターン（edge 439 ）の補間後の遷移図

Table 4.13: bit「0」を表すエッジパターン（edge 439 ）の補間後の遷移数

入力＼出力	438	439	447	511	合計
439	66	1401	147	1	1515
その他	0	8	0	0	8

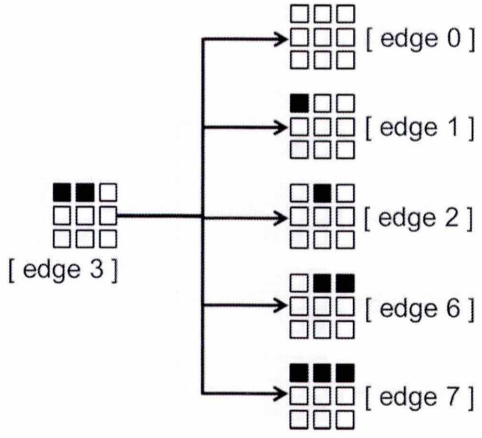


Fig. 4.21: bit「1」を表すエッジパターン（edge 3 ）の補間後の遷移図

Table 4.14: bit「1」を表すエッジパターン（edge 3 ）の補間後の遷移数

入力＼出力	0	1	2	3	6	7	合計
3	65	425	57	2818	1	33	3399
その他	0	0	0	145	0	0	145

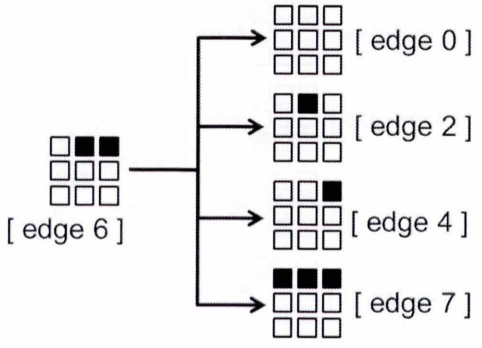


Fig. 4.22: bit「1」を表すエッジパターン（edge 6 ）の補間後の遷移図

Table 4.15: bit「1」を表すエッジパターン（edge 6 ）の補間後の遷移数

入力＼出力	0	2	4	6	7	合計
6	56	30	541	3132	16	3775
その他	0	0	0	156	0	145

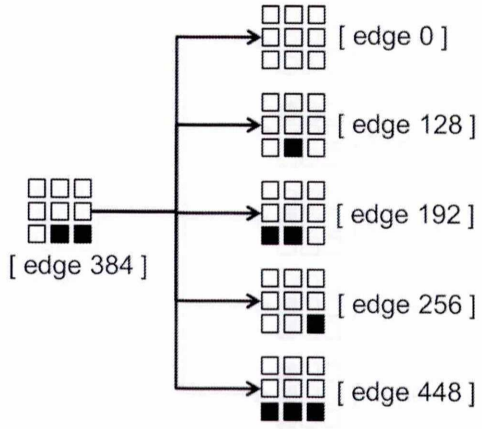


Fig. 4.23: bit「1」を表すエッジパターン（edge 384 ）の補間後の遷移図

Table 4.16: bit「1」を表すエッジパターン（edge384 ）の補間後の遷移数

出力 / 入力	0	128	192	256	384	448	合計
384	28	74	4	579	3182	48	3915
その他	0	0	0	0	71	0	71

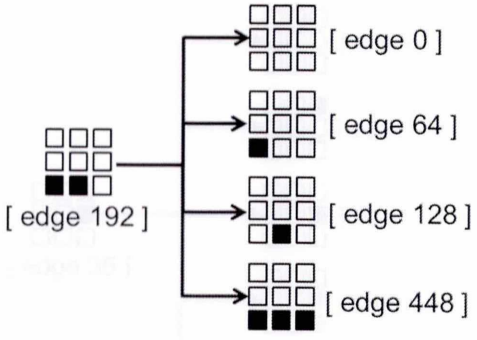


Fig. 4.24: bit「1」を表すエッジパターン（edge 192 ）の補間後の遷移図

Table 4.17: bit「1」を表すエッジパターン（edge 192 ）の補間後の遷移数

出力 / 入力	0	64	128	192	448	合計
192	60	543	41	3337	31	4012
その他	0	0	0	192	0	71

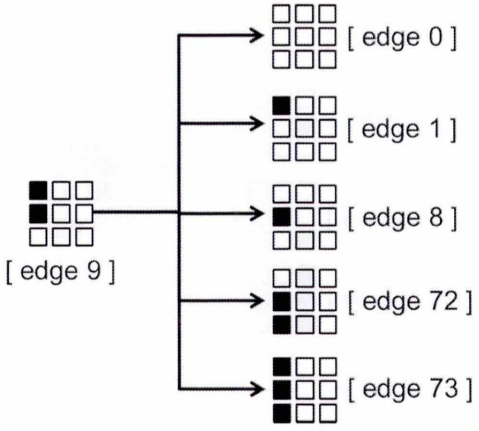


Fig. 4.25: bit「0」を表すエッジパターン（edge 9 ）の補間後の遷移図

Table 4.18: bit「0」を表すエッジパターン（edge 9 ）の補間後の遷移数

出力 / 入力	0	1	8	9	72	73	合計
9	26	403	63	2878	4	25	3399
その他	0	0	0	38	0	0	38

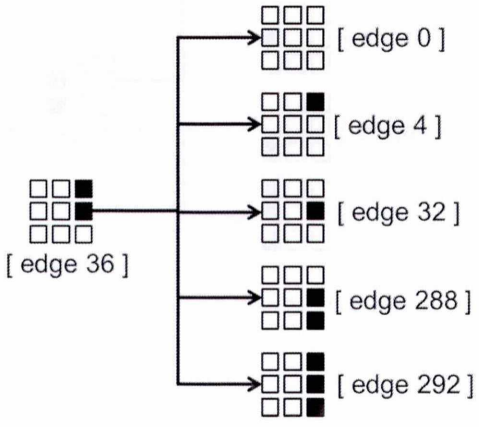


Fig. 4.26: bit「0」を表すエッジパターン（edge 36 ）の補間後の遷移図

Table 4.19: bit「0」を表すエッジパターン（edge 36 ）の補間後の遷移数

出力 / 入力	0	4	32	36	288	292	合計
36	26	422	42	3242	7	36	3775
その他	0	0	0	64	0	0	64

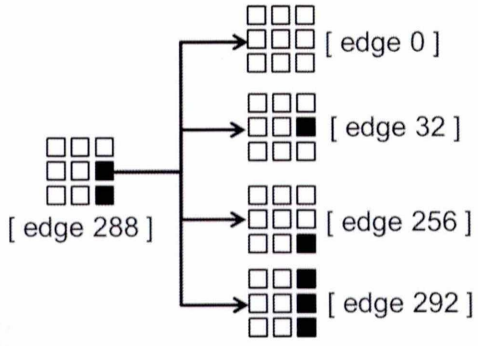


Fig. 4.27: bit「0」を表すエッジパターン（edge 288 ）の補間後の遷移図

Table 4.20: bit「0」を表すエッジパターン（edge 288 ）の補間後の遷移数

入力＼出力	0	32	256	288	292	合計
288	57	65	353	3418	22	3915
その他	0	0	0	37	0	37

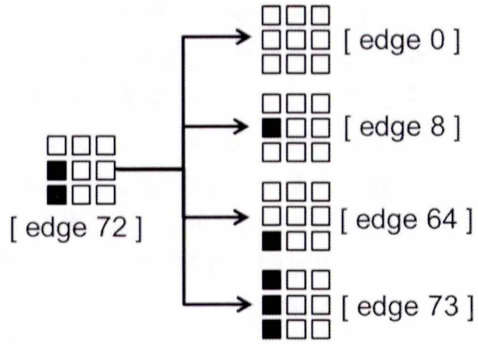


Fig. 4.28: bit「0」を表すエッジパターン（edge 72 ）の補間後の遷移図

Table 4.21: bit「0」を表すエッジパターン（edge 72 ）の補間後の遷移数

入力＼出力	0	8	64	72	73	合計
72	31	67	338	3543	33	4012
その他	0	0	0	75	0	75

第5章　考察

5.1　情報量の増加

本研究では、埋め込める情報量を増やすために従来のエッジパターンに加えて、7パターン計44通りのエッジパターンを提案した。従来のエッジパターンと新たに提案したエッジパターンを組み合わせ、画像に情報を埋め込むことで画像の劣化と情報量の増加を調べ実用性の可否を判断した。情報量を増やすために全パターンを使用して埋め込んだFig.4.6からFig.4.10の画像では等倍のままだと画質の劣化はほとんど見られないので、画像の劣化度合いを比較しやすくするために三つの拡大画像をFig.4.11に示した。(a)、(b)、(c)それぞれ原画像、既存のエッジパターンと、全パターンの拡大画像である。既存のエッジパターンによって情報を埋め込んだ画像は画質の劣化がほとんど目立たず、どこに情報が埋め込まれているか分からない。しかし全パターンを使用した拡大画像は劣化が顕著に出ている。既存のパターンのみを使用した埋め込みでは情報量が足らず、全パターンを使用した埋め込みでは拡大したときに劣化が目立ってしまう。そこで画質の劣化を抑えつつ情報を埋め込むためにいくつかのパターンを選び、組み合わせて埋め込みを行った。同じように三つの拡大画像をFig.4.12に示す。(a)、(b)、(c)それぞれ既存のパターン、全パターンと、推奨パターンを使用して埋め込んだ拡大画像である。推奨パターンは4章でも述べたが、既存のパターンとパターンB,D,Fの四つのパターンでできている。推奨パターンでの埋め込みは全パターンを使った埋め込みより劣化が少ないことがわかる。また、従来の既存のパターンの埋め込みと比較するとほとんど変わらずに埋め込みができていることから劣化を抑えることができた。しかし情報量の増加はどうだろうか。情報量の増加倍率を比較するために二つの表を用意した。Table 4.1とTable 4.2である。Table 4.1の既存パターンと全パターンの情報量の増加倍率は平均で約5倍の増加に成功しているものの全パターンを使用して情報の埋め込みを行うと劣化の比較からわかるようにあまり多用ができない。Table 4.2の既存パターンと推奨パターンの情報量の増加倍率は平均で約2倍の増加に成功した全パターンと比べると情報量の増加は少ないものの劣化を抑えて埋め込み情報量を増やすことができる。まとめるとすべてのエッジパターンを使用した情報の埋め込みは等倍の画像であるとほとんど劣化は目立たないが拡大すると画像の劣化が顕著に表れる。既存パターンと全てのパターンの情報量の差を比べてみると、従来の方法より約5倍の情報量を埋め込むことが可能である。これはビット数で表すと35,000ビット近く平均して増えており、従来のものより文字にして4,400文字多く埋め込めることができる。このことから埋め込みたい情報量が多い場合は提案したすべてのパターンを使用することが良いと考えられる。推奨パターンを使用した情報の埋め込みは、拡大画像ではまったく劣化が目立たない。従来のものより約2倍の情報量を埋め込めることがわかった。全パターンを使った埋め込みより少ないが増加していることに間違いはない。この組み合わせは、短く単発な情報を埋め込むときに使用できると考えられる。

5.2 誤り訂正符号を用いた位置ずれの検出

エッジパターン法の位置ずれに対応するするために、誤り訂正符号を用いた。誤り訂正符号を付加し、スキャンした位置の周りの 8 ピクセルからもスキャンし、合計 9 通りの誤りビット数を比較することで、最も誤りビット数の少ない位置を検出し、正しい読み取り開始位置と特定する。Table 4.3 の結果を見ると、本来のスキャン開始位置である”a”の位置から読み取った際の誤りビット数が 0 となっており正しい位置を特定できていることが分かる。位置”a”以外の位置にも誤りビット数が少ない位置があるが、これは埋め込んでいる情報が「0」か「1」のどちらかであるため、位置ずれを起こしても偶然「0」または「1」が一致した為だといえる。さらに本研究では (16,8) 符号を使用しているが、これは (16,8) は 2 ビットまでのランダム誤りと長さ 3 ビットまでのバースト誤りを訂正することができる。もちろん用途によって冗長度を変えても良いと考えられる。

5.3 ハフ変換を用いた傾き検出

スキャン時の傾きを検出するために、ハフ変換を用いて画像中の行から直線を複数検出しその直線からタンジェントの値を計算することで傾きを求める。その傾きを求める方法を使用し、検出した角度の比較を行った。今回は 200dpi の A4 サイズ 1654×2239 を使用しているため、許容誤差は誤り訂正符号を使用を考えない場合は約 0.035 度となる。許容誤差とは、エッジパターン法で情報を読み込むときに 1 ピクセル分画像が傾くだけで情報が読み込めなくなる。それならば画像の傾きを 1 ピクセル分に抑えなければいけないことから 1 ピクセル分の傾きを求めると 0.035 度となる。実験では 100 枚の画像を指定の傾きに傾け検出を行った。Table.2 では 0.1 度のときの結果はよくないものの、それ以降の傾きに対して正常に検出できているといえる。今回、ハフ変換に加えモロフォロジー変換を使用し文字にダイレーション処理を行ってからハフ変換を用いた傾き検出を行ったが結果は変わらずであった。今後、0.1 度以内の傾きに対しても正確に傾き検出を行えるように改良する必要がある。

5.4 埋め込み情報の読み取り

前述の手法により検出した傾をもとに画像を補正するために Bi-linear 法を利用した。Bi-linear 法を 2 値文書画像に使用すると 2 値ではなくなるため、回転した後に閾値 128 で 2 値化を行った。また、画像補間後は完全に同じ画像には戻らない。これは角度が大きくなるほど、誤差が増えていく。傾きが小さいときは誤差が少く、情報が失われていない箇所があることから URL 程度の短い情報を埋め込み情報として画像内に連続で埋め込むことで情報の読み取りが可能であると考えらる。しかし、傾きが大きいときは誤差が多く同様の方法では正常に情報を読み取れず、誤り訂正符号を付加した情報でも誤り訂正が行えないことから正しい情報を読み取れない。そこで、補間画像と原画像を比較し画像間でエッジパターンの誤差の出方に規則性がないか調べた。これらによると、3×3 の四隅の黒くなっている箇所を中心に複数のエッジパターンに遷移したことがわかる。この状態遷移表をもとに遷移の発生確率を組み合わせることで正しい情報を取り出すことが可能だと考えられる。

第6章　まとめ

本研究では、従来のエッジパターン法に比べて三つの部分が改善された。
一つ目は情報量の増加である。少なくとも 2 倍の情報量を埋め込むことができるようになった。また、できるだけ多くの情報量を埋め込みたい場合には約 4 倍の情報量を埋め込めるようになった。これは文字にすると 4,400 文字近い計算になる。

二つ目はスキャン時の位置ずれに対応できるようになった。これは誤り訂正符号を埋め込むことによって正しい位置を特定する方法だ。注目ピクセルとその周り 8 ピクセルから読み込むことによって誤りビット数を検出し、誤りビット数が 0 の位置を正しい読み取り開始位置として推定できるようになった。

三つ目はスキャン時の傾き検出である。これは画像中の行からハフ変換を用いて直線を検出しその直線をもとに傾きを検出するものである。この手法により非可視性を保ったまま傾きを検出することができたが、より高い精度で角度を検出できるように改良が必要である。

最後に改善案として、画像補間後の情報読み取り時にエッジパターンの壊れ方を調べた。この状態遷移表をもとにエッジパターンの組み換えを行うことで正しい情報を取り出せることが可能だと考えられる。

以上の三点より、従来のエッジパターン法より多くの情報量の埋め込みが可能となり、弱点であった位置ずれと傾きに対応できるようになった。今後は、本研究により調べた状態遷移表をもとに情報読み取りの実現可能を示唆した。

謝辞

本研究を進めるにあたり、本研究を進めるにあたり、彌冨仁准教授にはいつもあたたかくご指導、ご鞭撻を賜り、心からの感謝の念を表して謝辞とさせていただきます。

参考文献

[1] 今井秀樹, ”符号理論”, 電子情報通信学会, ISBN4-88552-090-8

[2] Brassil, J., Low, S., Maxemchuk, N., and O’Gorman, L., ”Electronic marking and identification techniques to discourage document copying”, IEEE J. Select. Arras Commun., 13, 1495, 1995.

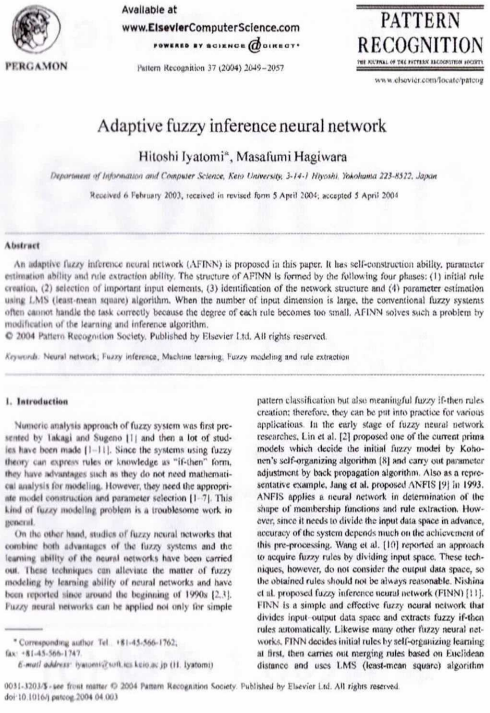
[3] 須崎　昌彦, 須藤　正之,”印刷文書への透かし埋込および検出方法”, 電子情報通信学会論文誌 A Vol.J87-A, No.6, pp.778-786, 2004.

[4] 阿部　悌、井上　浩一: ”２値画像への電子透かし”，Ricoh technical report，2000

付録

1. 画像内に特徴点を加えた検出方法

エッジパターン法は3× 3ピクセルごとにスキャンし、文字のエッジ部分のみに情報を埋め込む方法である。そのため、埋め込まれた情報は位置に依存するので、埋め込まれた情報を読み取るためスキャンする際に少しでも位置がずれてしまったり、傾いてしまうと正しい情報を取り出すことができない。そこで画像内の文書を囲むように特徴点となる四つの括弧を置く。その四つうち一つの括弧と隣り合わせになる括弧を検出することで横にずれた長さ、縦にずれた長さかタンジェントを求め角度を求め、それぞれの角度によるタンジェントの値と比較することで、傾いた角度を求める。Fig.6.1に実際に特徴点を加えた画像を示す。



PERGAMON

Available at www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Pattern Recognition 37 (2004) 2049–2057

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

Adaptive fuzzy inference neural network

Hitoshi Iyatomi*, Masafumi Hagiwara
Department of Information and Computer Science, Keio University, 3-14-1 Hiyoshi, Yokohama 223-8522, Japan

Received 6 February 2003; received in revised form 5 April 2004; accepted 5 April 2004

Abstract

An adaptive fuzzy inference neural network (AFINN) is proposed in this paper. It has self-construction ability, parameter estimation ability and rule extraction ability. The structure of AFINN is formed by the following four phases: (1) initial rule creation, (2) selection of important input elements, (3) identification of the network structure and (4) parameter estimation using LMS (least-mean square) algorithm. When the number of input dimension is large, the conventional fuzzy systems often cannot handle the task correctly because the degree of each rule becomes too small. AFINN solves such a problem by modification of the learning and inference algorithm.

Keywords: Neural network; Fuzzy inference; Machine learning; Fuzzy modeling and rule extraction

1. Introduction

Numeric analysis approach of fuzzy system was first presented by Takagi and Sugeno [1] and then a lot of studies have been made [1–11]. Since the systems using fuzzy theory can express rules or knowledge as "if-then" form, they have advantages such as they do not need mathematical analysis for modeling. However, they need the appropriate model construction and parameter selection [1–7]. This kind of fuzzy modeling problem is a troublesome work in general.

On the other hand, studies of fuzzy neural networks that combine both advantages of the fuzzy systems and the learning ability of the neural networks have been carried out. These techniques can alleviate the matter of fuzzy modeling by learning ability of neural networks and have been reported since around the beginning of 1990s [2,3]. Fuzzy neural networks can be applied not only for simple pattern classification but also meaningful fuzzy if-then rules creation; therefore, they can be put into practice for various applications. In the early stage of fuzzy neural network researches, Lin et al. [2] proposed one of the current prima models which decide the initial fuzzy model by Kohonen's self-organizing algorithm [8] and carry out parameter adjustment by back propagation algorithm. Also as a representative example, Jang et al. proposed ANFIS [9] in 1993. ANFIS applies a neural network in determination of the shape of membership functions and rule extraction. However, since it needs to divide the input data space in advance, accuracy of the system depends much on the achievement of this pre-processing. Wang et al. [10] reported an approach to acquire fuzzy rules by dividing input space. These techniques, however, do not consider the output data space, so the obtained rules should not be always reasonable. Nishina et al. proposed fuzzy inference neural network (FINN) [11]. FINN is a simple and effective fuzzy neural network that divides input–output data space and extracts fuzzy if-then rules automatically. Likewise many other fuzzy neural networks, FINN decides initial rules by self-organizing learning at first, then carries out merging rules based on Euclidean distance and uses LMS (least-mean square) algorithm

* Corresponding author. Tel.: +81-45-566-1762; fax: +81-45-566-1747.
E-mail address: iyatomi@soft.ics.keio.ac.jp (H. Iyatomi)

0031-3203/$ - see front matter © 2004 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2004.04.003

Fig. 6.1: 画像に特徴点を加えた画像

2. スキャン時の傾き検出

スキャン時の傾きに対応するために画像内に可視情報を加え角度検出を行った。実際に 0.0 度から 1.0 度まで傾いた画像を Fig.6.2 に示す。図からわかるように人は画像が 1 度傾いているだけで違和感を感じることができるこのことからそれ以上の傾きの検出は見ただけで修正できるので 1.0 度以内での認知しにくい傾きの検出を行い、傾いた画像から求めたタンジェントの値と検出した角度を示す。Table2 にタンジェントの値と実際に求めた角度の検出結果を示す。画像から情報を正常に読み取る時に、1 ピクセルずれてしまうと読み取れなくなってしまうのでこの範囲にズレを収めたい。なので 1 ピクセル分の傾きを許容誤差角度として求めると 0.0231 度となり検出した傾きをこの範囲以内で収めたい。実際に角度をつけた画像からはタンジェントを検出でき、求めた角度実際の角度のと誤差は許容誤差以内であるため 1.0 度までの傾きに対して正常に傾き検出検出できた。

Fig. 6.2: 三つの傾きを比較する画像

3. スキャン時の傾き検出 - 考察 -

スキャン時の傾きを検出するために、可視情報として画像の四隅に特徴点となる括弧上の印を加え、その傾きを求める方法を使用し、検出した角度の比較を行った。今回は 300dpi の A4 サイズ (2480 × 3058) を使用しているため、許容誤差は誤り訂正符号を使用を考えない場合は約 0.0232 度となる。許容誤差とは、エッジパターン法で情

Table 6.1: それぞれの角度に対応するタンジェントの値

角度	タンジェントの値
0.0	0.00000
0.1	0.00174
0.5	0.00872
1.0	0.01745

Table 6.2: 検出したタンジェントの値と傾きの角度と実際の角度との誤差

画像の傾き	検出したタンジェント	求めた角度	実際の角度との誤差
-1.0	0.01709	0.97943	0.02058
-0.5	0.00841	0.48216	0.01784
-0.1	0.00157	0.09018	0.00982
0.0	0.00000	0.00000	0.00000
0.1	0.00158	0.09018	0.00982
0.5	0.00842	0.48216	0.01784
1.0	0.01709	0.97943	0.02057

報を読み込むときに１ピクセル分画像が傾くだけで情報が読み込めなくなる。それならば画像の傾きを１ピクセル分に抑えなければいけないことから１ピクセル分の傾きを求めると 0.0232 度となる。傾きの検出するときは画像を読み込むときに四隅の括弧を検出する。検出できた四つ括弧の角の座標を基準として、対角の位置を除いて二つの点で作れるタンジェントの値を四つ計算した。対象画像１枚で-1 度から１度までの計７回の検出を行った結果全ての傾きで実際との傾き誤差が許容誤差以内であり、次の段階である傾きの補正を行っても正しく読み取れると考えられることがわっかり、正しく検出できた。

発表論文

1. 2012年9月　第11回情報科学技術フォーラムにて発表（演題：改良エッジパターン法による文書画像の電子透かし）