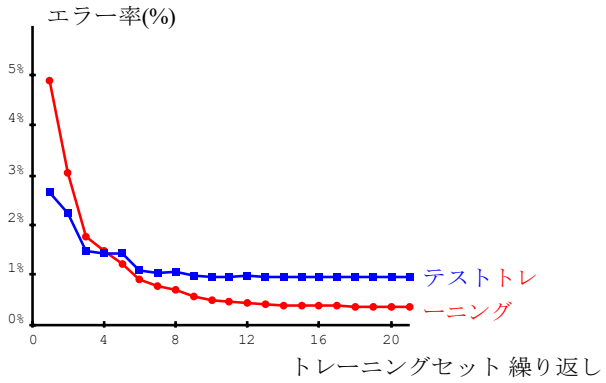


図5.LeNet-5の訓練誤差とテスト誤差を、6万パターンの訓練セット（歪みなし）を通過した回数の関数として示したもの。平均的な学習誤差は、学習が進むにつれてその場で測定される。このため、学習誤差がテスト誤差より大きく見える。10〜12回学習セットを通過すると収束する。

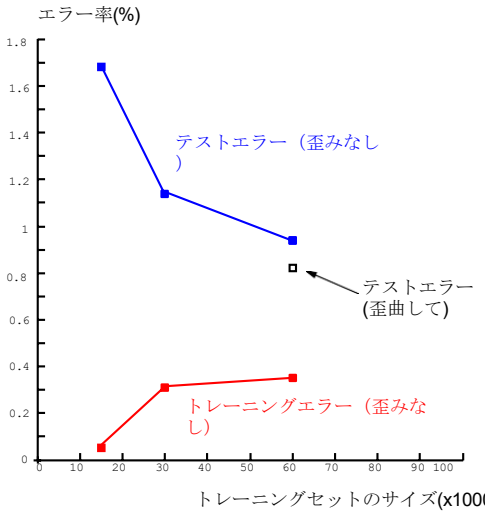


図6.様々なサイズのトレーニングセットを用いて達成されたLeNet-5のトレーニング誤差とテスト誤差。このグラフから、より大きな学習セットを用いることで、LeNet-5の性能が向上することがわかる。また、空洞の四角は、ランダムな歪みを用いて人為的に多くの学習パターンを生成した場合のテスト誤差を示す。テストパターンは歪んでいない。

歪みパラメータをランダムに選び、歪んだパターンを作成した。歪みは次の平面アフィン変換の組み合わせである：水平・垂直移動1拡大縮小1スクイーズ（水平圧縮と垂直伸長1またはその逆）1および水平剪断である。図7は、学習に使用した歪みパターンの例である。歪んだデータをトレーニングに使用した場合1、テストエラー率は0.8%に減少した（変形なしの場合0.95%）。学習パラメータは変形なしと同じものを使用した。また、学習セッションの長さは変更しない（601000パターン×20パス）。この20回の学習で、ネットワークは各サンプルを2回しか見ていないことは興味深

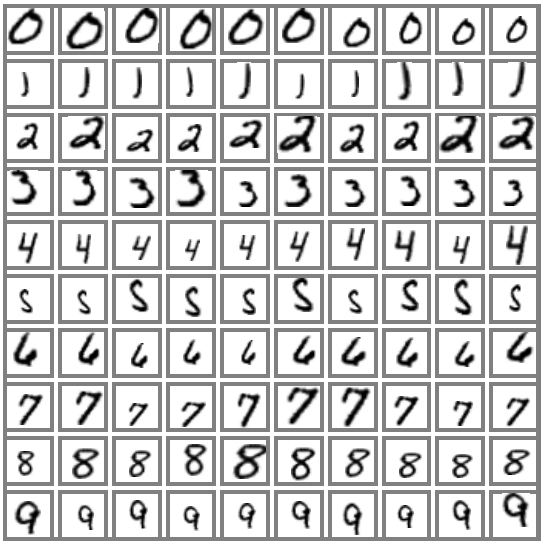
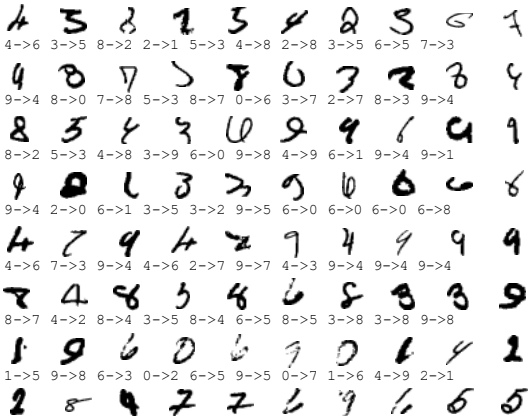


図7.　10パターンのトレーニングの歪みの例。



い。

図8は、82個の誤分類されたテスト例を示している。これらの例のいくつかは、本当に曖昧であるが1、いくつかの例は

2->8 8->5 4->9 7->2 7->2 6->5 9->7 6->1 5->6 5->0
4->9 2->8

図8.LeNet-5が誤分類した82のテストパターン。各画像の下に正解（左）、ネットワークの誤答（右）を表示。これらの誤りの多くは、本当にあいまいなパターンや、学習セットで表現が不十分なスタイルで書かれた数字が原因である。

を完全に識別することが可能である。このことから、今後、学習データを増やすことで、さらなる改善が期待される。

C.他の分類器との比較

比較のために1，同じデータベースに対して，他の様々な学習可能な分類器を学習・テストしました．これらの結果の初期のサブセットは[51]で発表されています．様々な手法のテストセットにおけるエラー率を図9に示す．

C.1 線形分類器1 およびペアワイズ線形分類器

最も単純な分類器として考えられるのは，次のようなものです．

線形分類器。各入力画素値は、各出力ユニットの加重和に寄与する。最も高い和を持つ出力ユニット（バイアスの寄与を含む）は、その出力ユニットに属する。

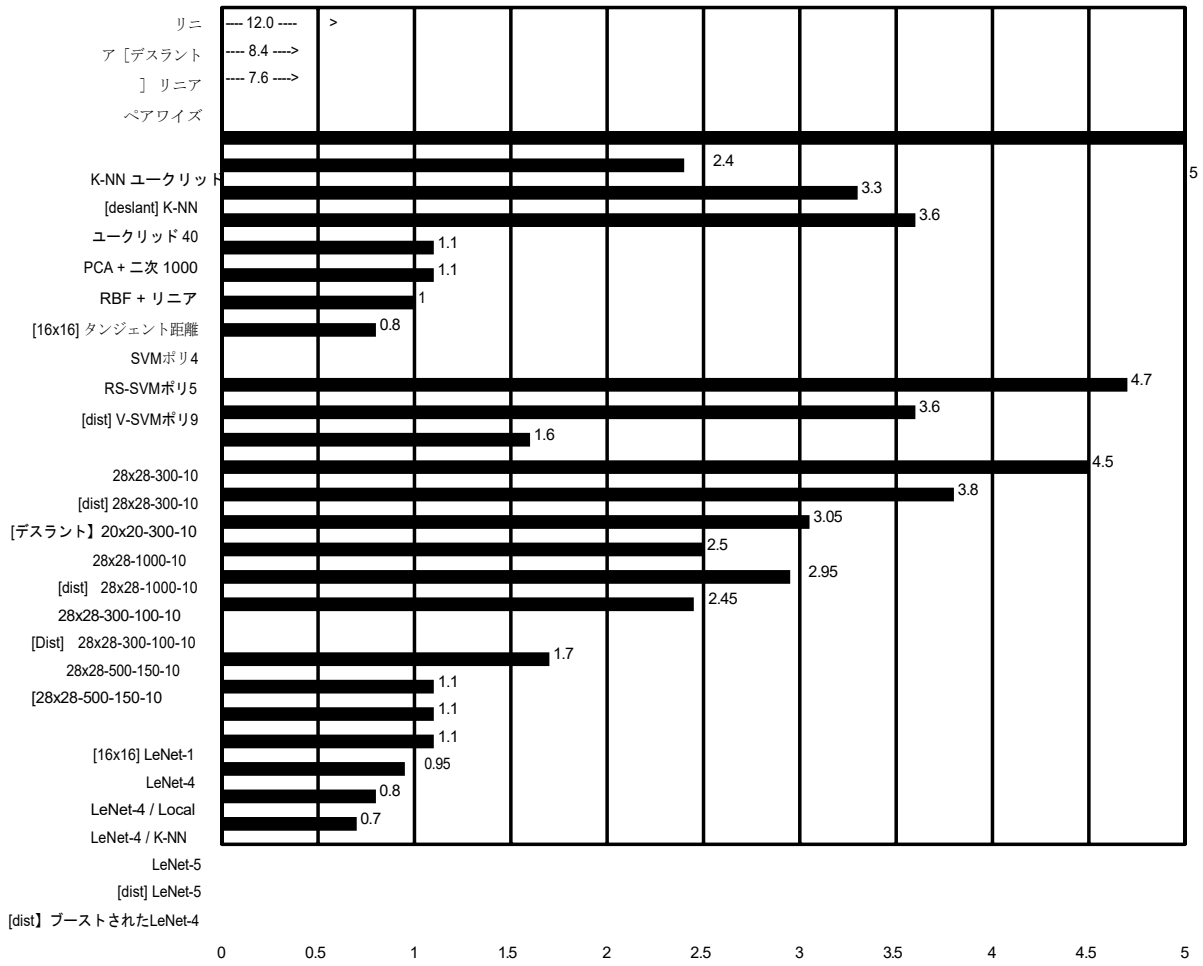


図 9. 様々な分類法におけるテスト集合の誤り率（%）. deslant] は分類器がデータベースのデスラントバージョンで学習・テストされたことを示す. dist] は学習集合が人為的に歪められた例で補強されたことを示す. 16x16] は, システムが16x16ピクセルの画像を使用したことを示します. 引用したエラー率の不確かさは約0.1%である。

stant)は、入力文字のクラスを示す。通常のデータ1では、誤り率は12%である。ネットワークは7850個の自由パラメータを持つ。デスラントイメージ1では、テストエラー率は8.4%で、ネットワークには4010の自由パラメータがあります。線形分類器の欠陥はよく知られており[1]，ここでは単に，より洗練された分類器との比較の基礎とするために，線形分類器を含めています．シグモイドユニット、線形ユニット、勾配降下学習1、線形システムを直接解くことによる学習など、様々な組み合わせで同様の結果を得ることができました。

基本的な線形分類器を単純に改良したものがテストされました[52]．このアイデアは，各クラスを他のクラスから分離するために，単層ネットワークの各ユニットを訓練することです．我々の場合、この層は 0/11 0/21...0/91 1/2...8/9 とラベル付けされた45ユニットで構成される。ユニットi/jはクラスi1のパターンで+1、クラスj1のパターンで-1を生成するよう学習され、他のパターンでは学習されない。クラスiの最終スコアは，すべてのxとyについて，i/xとラベル付けされたすべてのユニットの出力の合計から，y/i1とラベル付けされたすべてのユニットの出力の合計を差し引いたものである．

C.2 ベースラインニアレストネイバ分類器

もうひとつの簡単な分類器は，入力画像間のユークリッド距離を用いた K-nearest neighbor 分類器である．この分類器は，学習時間1や設計者の頭脳1が不要であるという利点があります.

しかし、20×20ピクセルの学習用画像601000枚（1ピクセル1バイトで約24メガバイト）を実行時に使用しなければならないため、メモリ容量と認識時間が大きくなってしまう。もっとコンパクトな表現にすれば、誤認識率をそれほど上げずに済むかもしれない。通常のテストセットでは、誤差は5.0%であった。デスランテッドデータ1では、kが3 の とき、誤差は2.4%1であった。現実的なユークリッド距離最近傍探索システムは、画素を直接処理するのではなく、特徴ベクトルを処理するが1、本研究で紹介した他のシステムはすべて画素を直接処理しているので1、この結果は基準比較として有用である。

C.3 主成分分析(PCA)と多項式分類器

53]1 [54]1 に従い，入力パターンを学習ベクトル集合の40主成分に投影する前処理段階を構築した．主成分を計算するために，まず各入力成分の平均が計算され，学習ベクトルから差し引かれた．そして，得られたベクトルの共分散行列を計算し，特異値分解を用いて対角化した．40次元の特徴ベクトルは，2次多項式分類器の入力として使用された．この分類器は，821個の入力を持つ線形分類器と見なすことができます1.

を入力変数の組の積で表現した。通常のテストセットでの誤差は3.3%であった。

C.4 Radial Basis Function ネットワーク

55]1 に従って、RBFネットワークを構築した。第1層は28×28入力の11000ガウス型RBFユニット1、第2層は単純な1000入力の / 10出力の線形分類器である。RBFユニットは100個ずつ10グループに分けられた．各グループは、適応K-meansアルゴリズムを用いて、10クラスのうちの1クラスのすべての学習例について学習された。第2層の重みは正則化された擬似逆行法によって計算された。通常のテストセットにおける誤差は3.6%であった。

C.5 1層隠れ家型完全連結多層ニューロネットワーク

もうひとつの分類器は，付録 Cで説明したバックプロパゲーションを用いて学習させた，2層の重み（隠れ層）を持つ完全連結型多層ニューラルネットワークをテストしました．通常のテストセットにおける誤差は，300の隠れユニットを持つネットワークで4.7%，1000の隠れユニットを持つネットワークで4.5%であった1．人工的な歪みを用いてより多くの学習データを生成したところ、わずかな改善しか得られなかった。300の隠れユニット1では3.6%、1000の隠れユニットでは3.8%であった。デスランテッド画像を使用した場合1、テスト誤差は300の隠れユニットのネットワークで1.6%に減少しました。

このように多くの自由パラメータを持つネットワークが、合理的に低いテスト誤差を達成することができるのは、依然として謎である。我々は、多層網における勾配降下学習のダイナミックスに「自己正則化」効果があると推測している。重み空間の原点はほとんど全ての方向に魅力的な鞍点であるため1、最初の数エポックでは必ず重みが小さくなる（最近の理論解析ではこれを確認しているようである[56]）。重みが小さいとシグモイドは準線形領域で動作することになり1、このネットワークは本質的に低容量1単層ネットワークと同等となる。学習が進むと1、重みが大きくなり1、ネットワークの実効容量が徐々に大きくなる。これは、Vapnikの「構造的リスクの最小化」原理[6]を、偶然とはいえほぼ完璧に実装したものであるように思われる。これらの現象に対するより良い理論的理解1 とより多くの実証的証拠1が必要であることは間違いない。

C.6 2層隠れ家型完全連結多層膜ニューラルネットワーク

このアーキテクチャの効果を見るために、2つの隠れ層からなる多層ニューラルネットがいくつか学習された。理論的な結果から、どのような関数も1層のニューラルネットで近似できることが示されている[57]。しかし、2つの隠れ層から構成されるニューラルネットワークの方が、実用的な場面でより良い性能を発揮することがあることが、複数の著者によって示されている。この現象はここでも観察された。28x28-300-100-10 ネットワークのテストエラー率は 3.05%1であり、重みや接続をわずかに増やした 1階層ネットワーク1
よりもはるかに良い結果であった。さらに、28x28-1000-150-10のネットワークでは

は、エラーレートがわずかに改善されただけでした。2.95%.歪んだパターンを用いた学習では、28x28-300-100-10ネットワーク1では2.50%、28x28-1000-150-10ネットワークでは2.45%と、若干の改善がみられました。

C.7 小さな畳み込みネットワーク。LeNet-1畳み込みネットワークは，この問題を解決するための試みである。
学習セット1
を学習できない小さなネットワークと、パラメータ化されすぎた大規模なネットワークとの間のジレンマ。LeNet-1はConvolutional
Networkアーキテクチャの初期の実装であり、ここでは比較のために含まれている。画像は16x16ピクセルにダウンサンプリングされ、28x28の入力層の中央に配置された。LeNet-
11の評価には約1001000の乗算/加算ステップが必要ですが、畳み込み処理の性質上、自由パラメータ数は約2600にとどまります。 LeNet-11の
1のアーキテクチャは、USPS（US Postal Service zip codes）データベースの独自版を用いて開発され、そのサイズは利用可能なデータに合わせて調整された[35]。LeNet-
1は1.7%のテスト誤差を達成した。このようにパラメータ数の少ないネットワークでこのような良好なエラー率を達成できることは、このアーキテクチャが課題に適していることを示している。

C.8 LeNet-4

LeNet-
1の実験では、大規模な学習セットを最適に利用するためには、より大きな畳み込みネットワークが必要であることが明らかになった。LeNet-
4とその後のLeNet-
5 は、 この問題に対処するために設計されました。LeNet-4は、アーキテクチャの詳細を除いて、LeNet-
51と非常によく似ている。
4つの第1レベル特徴マップ1と、それに続く
8個のサブサンプリングマップが各第1層の特徴マップに対で接続され、次に16個の特徴マップ1、16個のサブサンプリングマップ1が続き、完全接続層には
120ユニット1、出力層(10ユニット)と続く。LeNet-
4は約2601000の接続を持ち、約171000の自由パラメータを持っている。テスト誤差は1.1%であった。一連の実験1では，LeNet-4の最終層をEuclidean Nearest Neighbor分類器1 とBottou and Vapnik [58]1の「局所学習」法（新しいテストパターンが示されるたびにlocal
linear分類器を再トレーニングする）に置き換えました．これらの方法は，棄却率を向上させたが，生の誤り率1 を向上させることはできなかった．

C.9 ブースト付きLeNet-4

R. Schapire [59]1 の理論的研究に続いて，Drucker ら[60]
は，複数の分類器を結合するための「ブースティング」手法を開発した．2番目のネットは，1番目のネットによってフィルタリングされたパターンで学習され，2番目のマシンには，1番目のネットが正解したパターン1
と間違ったパターン1
が50%ずつ混在するようにする．最後に3番目のネットは、1番目と2番目のネットが一致しない新しいパターンで学習される。テストでは、3つのネットの出力が単純に加算される1。LeNet-
4のエラーレートは非常に低い1ため、2番目と3番目のネットの学習に十分なサンプルを得るために、LeNet-
5と同様に人工的に歪んだ画像を使用する必要があった。テスト誤差

は0.7%1
であり，どの分類器よりも優れていた．一見すると，ブースティングは1つのネットの3倍のコストがかかるように見える．実際，最初のネットが高い信頼度の答えを出すと1，他のネットは呼ばれない．平均計算量は単一ネットの約1.75倍である．

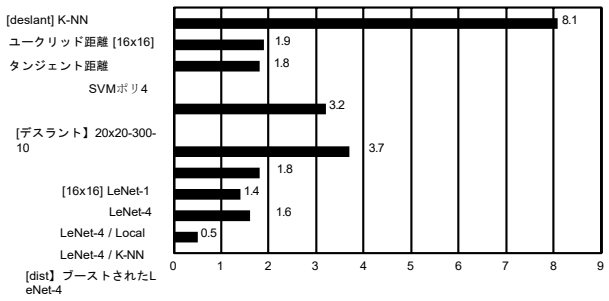
C.10 接線距離分類器(TDC)

TDC (Tangent Distance Classifier)
は，距離関数を近傍に配置した近傍分類法である．
は，入力画像の小さな歪みや平行移動に対して敏感である
[61]．画像を高次元画素空間（次元は画素数に等しい）1
の点と考えると，ある人物の進化する歪みは画素空間において曲線を描くことになる．これらの歪み1
をすべて合わせると、ピクセル空間に低次元の多様体が定義される。この多様体は、元画像の近傍の小さな歪み1 の場合、接平面と呼ばれる平面1で近似することができる。文字画像の「近さ」の優れた指標は、接平面1間の距離であり、平面を生成するために使用される一連の歪みは、並進1拡大縮小1斜め1絞り1回転1および線の太さの変化を含んでいる。16x16ピクセルの画像を用いたテストでは、1.1%の誤差を達成した。複数の解像度で単純なユークリッド距離を用いたプレフィルタリング技術により、必要な接線距離の計算回数を削減することができた。

C.11 サポートベクターマシン(SVM)

多項式分類器は、複雑な決定面を生成するための手法としてよく研究されている。しかし、残念ながら、高次元の問題では、積項の数が膨大になるため、実用的ではありません1。サポートベクター法は、多項式や他の多くの種類の曲面を含む高次元空間1
の複雑な曲面を表現する非常に経済的な方法である[6]。

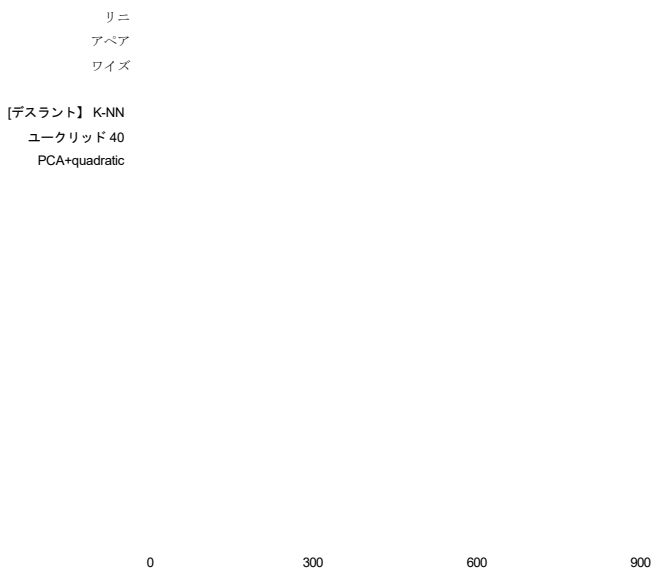
決定面の特に興味深い部分集合は，積項の高次元空間における2つのクラスの凸包から最大距離にある超平面に対応するものである．Boser1 Guyon1 と Vapnik [62]
は、この「最大マージン」集合における次数 k の任意の多項式は、まず入力画像と学習サンプルの部分集合（「サポートベクトル」と呼ぶ）1
の内積を計算し、その結果を k 乗1して得られた数を線形に結合すれば計算できることに気付いた。サポートベクターと係数を求めることは、線形不等式制約のある高次元二次最小化問題を解くことになる。比較のため1、Burges and Scholkopfが[63]で報告した結果をここに記す。通常のSVM1を用いた場合、通常のテスト集合におけるエラー率は1.4%であった。CortesとVapnikは同じデータに対して、少し異なる手法を用いたSVMで1.1%の誤差を報告している。この手法の計算コストは非常に高く、1回の認識で約1400万回の乗算加算が必



要である。ScholkopfのVirtual Support Vectors (V-SVM) では、1.0%の誤差を維持した。最近では1
Scholkopf (personal communication)

図10.

拒絶性能：いくつかのシステムにおいて、0.5%の誤差を達成するために拒絶しなければならないテストパターンの割合。



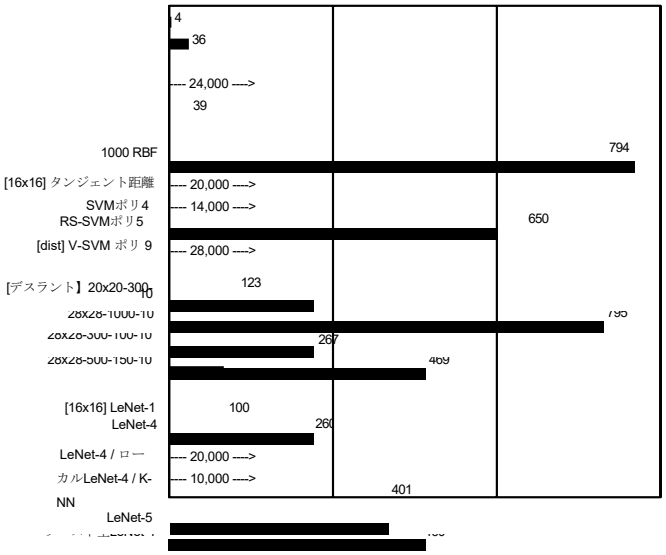


図11.サイズ正規化された画像から1文字を認識するための乗算累積演算数。

は、V-SVMの改良版を使って0.8%に到達しました。しかし、V-SVMは非常に高価であり、通常のSVMの約2倍である。この問題を解決するために、BurgesはReduced Set Support Vector technique (RS-SVM)1を提案し、通常のテストセットで1.1%を達成しました[63]1。

D.ディスカッション

分類器の性能の概要を図9から図12に示す．図 9 は101000
例のテストセットに対する分類器の生の誤り率である．ブーストされたLeNet-4は0.7%1という最も良い結果を示し、LeNet-5の0.8%がそれに続いた。

図10は，いくつかの手法において，誤差0.5%を達成するために棄却しなければならないテスト集合のパターン数を示している．パターンが棄却されるのは、応答出力の値があらかじめ定義された閾値より小さい場合である。多くのアプリケーションでは1、拒絶性能は生のエラーレートよりも重要である。パターン棄却の判断に用いられるスコアは、上位2クラスのスコアの差である。ここでも1 Boosted LeNet-4が最も良い性能を示した。LeNet-4の強化版は、生誤差が小さいにもかかわらず、オリジナルのLeNet-41よりも良い結果を示した。

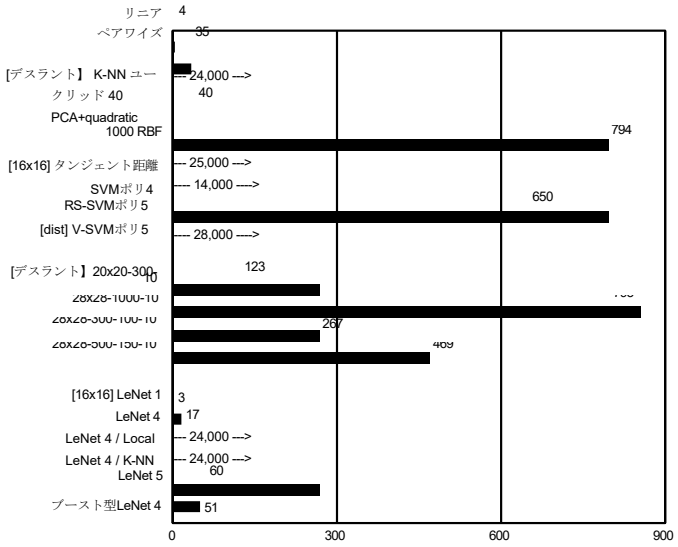


図 12. 各手法のメモリ要件（変数数で測定）。ほとんどのメソッドでは、1変数あたり1バイトで十分な性能が得られる。

の精度は同じでした。

図11は、各方式において、1枚のサイズ規格化画像を認識するために必要な積和演算の回数を示したものである。予想通り1
ニューラルネットワークは、メモリベース方式に比べ、はるかに負荷が低い。畳み込みニューラルネットワークは、その規則的な構造と重みのための必要メモリが少ないことから、特にハードウェア実装に適しています。LeNet-5の前身であるアナログ・デジタル混在のシングルチップ実装は、1000文字/秒を超える速度で動作することが示されている[64]。しかし、主流のコンピュータ技術の急速な進歩により、これらのエキゾチックな技術はすぐに時代遅れになってしまう。また、メモリベース技術のコスト効率の良い実装は、膨大なメモリ要件1
と計算要件により、より困難なものとなっています。

また、学習時間も測定した。K-ニアレストネイバーとTDCは基本的に学習時間がゼロであった。単層ネット1、ペアワイズネット1、PCA+2次ネットは1時間未満で学習できましたが、多層ネットの学習時間はもっと長いと予想されました1が、学習セットを通して10～20回パスする必要があるだけでした。これは、Sil-icon Graphics Origin 2000サーバー1上で200MHzのR10000プロセッサ1台を使い、LeNet-5を訓練するのに2～3日かかるCPU量に相当します。この学習時間は設計者にとっては重要ですが1、システムの最終的なユーザーにとってはほとんど意味がないことに注意してください。既存の技術1
と、かなりの訓練時間を犠牲にしてわずかな精度の向上をもたらす新しい技術1
のどちらかを選ぶとしたら、最終的なユーザーは後者を選ぶでしょう。

図 12は，記憶する必要のある変数の数で測定した，さまざまな分類法のメモリ要件1
およびしたがって自由パラメータ数1
を示している．ほとんどの手法は，適切なパフォーマンスを得るために，1 変数あたり約 1 バイトしか必要としません．しかし，ニアレスト・ネイバー法では，1 ピクセルあたり 4 ビットのメモリで十分な性能を発揮することができる．

をテンプレート画像に変換する。当然のことながら1
ニューラルネットワークは、メモリベースの方法よりもはるかに少ないメモリしか必要としません。

全体的な性能は、精度、実行時間、必要なメモリなど多くの要因に依存する。コンピュータ技術の進歩に伴い、より大容量の認識器が実現可能になった。より大きな認識器は、より大きな学習セットを必要とする。LeNet-1は1989年に利用可能な技術に適しており、LeNet-5は現在に適している。1989年当時、LeNet-5のような複雑な認識器は、数週間の学習が必要であり1、利用可能なデータ量を超えていたため、検討さえされなかった。長い間1 LeNet-1が最先端であると考えられていた。LeNet-1を模倣して，局所学習分類器1，最適マージン分類器1，接線距離分類器が開発され，成功した．しかし，その結果，より優れたニューラルネット・アーキテクチャの探索が動機づけられました．この探索は、学習例数の関数としての学習・テスト誤差の測定から得られた、様々な学習機械1の能力の見積もりから導かれたものであった。我々は、より大きな能力が必要であることを発見した。その結果、LeNet-4とLeNet-5を開発した。

ブースティングにより、メモリと計算機の費用を比較的抑えながら、精度を大幅に向上させることができることがわかった1。また、歪みモデルを用いることで、より多くのデータを収集することなく、データセットの有効サイズを増加させることができる1。

サポートベクターマシンは，他の高性能な分類器とは異なり，問題に対する先験的な知識を含んでいないため，優れた精度1を実現しています．実際1，この分類器は，画像のピクセルが固定マッピングで並べ替えられ，絵画的な構造が失われたとしても，同じように動作します．しかし、畳み込みニューラルネットワークに匹敵する性能に到達するには、メモリと計算機でかなりの犠牲を払わなければなりません。縮小版SVMの性能は畳み込みニューラルネットワークの2倍程度であり1、エラー率も非常に近い。この技術は比較的新しいものであるため、この結果の向上が期待される1。

多くのデータが利用可能な場合1、多くの手法で十分な精度を維持することができます。ニューラルネットの手法は、メモリベースの手法に比べて、実行速度がはるかに速く、必要なスペースもはるかに少なくて済みます。ニューラルネットの優位性は、学習用データベースの規模が大きくなればなるほど、より顕著になります。

E.不変性と耐ノイズ性

畳み込みネットワークは、実世界の文字列認識システムにおけるヒューリスティックセグメンテーションによって生成されるような、サイズ1 位置1および方向1
が大きく変化する形状を認識または拒否するのに特に適している。

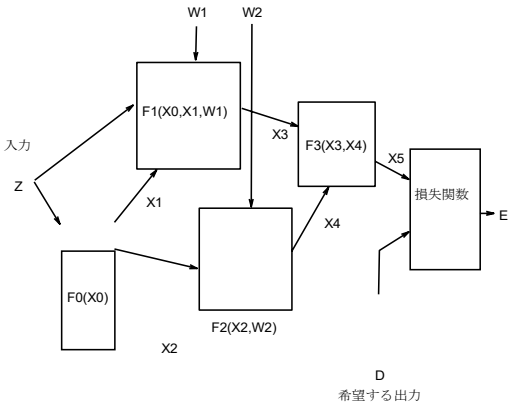
上記のような実験では、ノイズ耐性と歪み不変性の重要性は明らかではない。
実際のアプリケーションでは

とは全く異なります。一般に、文字は認識する前に文脈から切り離す必要があります。セグメンテーション・アルゴリズムが完璧であることはほとんどなく、文字画像に余計なマーク（ノイズ、アンダーライン、隣接する文字）1が残ったり、文字を切りすぎて不完全な文字ができたりすることがよくあります。このような画像は、サイズ正規化やセンタリングができない。不完全な文字を正規化することは、非常に危険なことです。例えば1
拡大された迷子マークは、本物のマークに見えることがある。そのため、多くのシステムでは、フィールドや単語のレベルで画像を正規化することが行われている。私たちの場合1、上下の
のプロファイルを検出し、一定の高さに正規化することで、フィールド全体（チェックの量）の画像を得ることができます。これにより、文字画像に浮き出ることはなくなるが1
、セグメンテーション後の文字の大きさや縦方向の位置のばらつきが大きくなる。そのため、このようなばらつきに強い認識器を使用することが望ましい。図13に、LeNet-
5で正しく認識された歪んだ文字の例を示す。スケール変動は約21分の1まで、縦方向のシフト変動は文字の高さの約半分まで1、回転は約30度まで正しく認識できると推定される。複雑な形状の完全な不変性認識はまだ達成されていないが1、畳み込みネットワークは、幾何学的な歪みに対する不変性または頑健性の問題に対する部分的な答えを提供すると思われる。

図13には、LeNet-
のロバスト性の例が含まれています。
5は、非常にノイズの多い条件下で処理されます。これらの画像を処理することは、多くの手法にとってセグメンテーションと特徴抽出の乗り越えられない問題を引き起こすが1、LeNet-
5はこれらの乱雑な画像から顕著な特徴を頑健に抽出することができるようである。このネットワークに使用された学習セットは、MNIST学習セットにソルト&ペッパーノイズを追加したものである。各画素は確率0.1で反転される。 LeNet-
5の他の動作例は、インターネット上のhttp://www.research.att.com/-yann/ocr で見ることができます。

IV. マルチモジュールシステムとグラフトランスネットワーク

前節で説明し使用した古典的なバックプロパゲーションアルゴリズム1
は、勾配学習の単純な形態である。しかし、式4で示される勾配逆伝播アルゴリズムは、線形トランスフォーム
とシグモイド関数の交互配列からなる単純な多層フィードフォワードネットよりも一般的な状況を記述していることは明らかである



1。原理的には、機能モジュール1
のヤコビアンの積を任意のベクトルで計算できるのであれば、どのような配列の機能モジュール1
でも導出はバックプロパゲートすることができる。なぜ、複数の異種モジュールで構成されるシステムを訓練したいのだろうか。その答えは、大規模で複雑な学習システムは、単純な1つの特化したモジュールから構築される必要があるからである。最も単純な例はLeNet-
51で、畳み込み層1、サブサンプリング層1、完全連結層1、RBF層が混在している。次の2つのセクションで説明する、あまり単純ではないもう一つの例1
は、「虹彩」を認識するためのシステムである。

図14 異種モジュールで構成される学習可能なシステム
異種モジュールで構成される学習可能なシステム。

を学習させることで、一度も正しいセグメンテーションを与えられずに、単語1の分割と認識を同時に行うことができる。

図 14 は，訓練可能なマルチモジュールシステムの一例である．マルチモジュールシステムは，各モジュールが実装する機能1 と，モジュール間の相互接続のグラフによって定義される．グラフは暗黙のうちに、フォワードパスでモジュールが更新されなければならない部分的な順序を定義している。たとえば、図 141 では、まずモジュール 0 が更新され1 、次にモジュール 1 と 2 が（おそらく並行して）更新され1 、最後にモジュール 3 が更新されます。モジュールには，学習可能なパラメータがある場合とない場合がある．システム1 の性能を測定する損失関数1は、モジュール4として実装される。最も単純なケース1 では、損失関数モジュールは、所望の出力を持つ外部入力を受け取る。この場合、学習可能なパラメータ（図中のW1、W2）1 外部入出力（Z、D、E）1 と中間状態変数（X1、X2、X3、X4、X5）の間に質的な差はない。

A.オブジェクト指向のアプローチ

オブジェクト指向プログラミングは、マルチモジュールシステムを実装する上で特に便利な方法である。各モジュールは、あるクラスのインスタンスである。モジュールクラスは、モジュールの入力と出力を引数とする fprop と呼ばれる "for-ward propagation" メソッド（またはメンバ関数）を持っている。例えば、図14のモジュール3の出力を計算するには、モジュール3のメソッドfpropを引数X3,X4,X5で呼び出すことで可能である。複雑なモジュールは、単純なモジュールから新しいクラスを定義することによって構築される。そのクラスのスロットには、メンバーモジュール とそれらのモジュール間の中間状態変数が含まれる。このクラスのfpropメソッドは、適切な中間状態変数や外部入出力を引数として、メンバーモジュール1 のfpropメソッドを呼び出すだけである。このアルゴリズムは、影響グラフがサイクルを持つようなモジュール1 のネットワークにも容易に一般化できるが、ここでは有向無サイクルグラフ（フィードフォワードネットワーク）の場合に限定して考察を行う。

多モジュールシステムにおける導関数の計算も同様に簡単である。そのために、各モジュールクラスのbprop1という「逆伝播」メソッド1 を定義することができる。モジュールの bprop メソッドは、同じ ar-をとる。

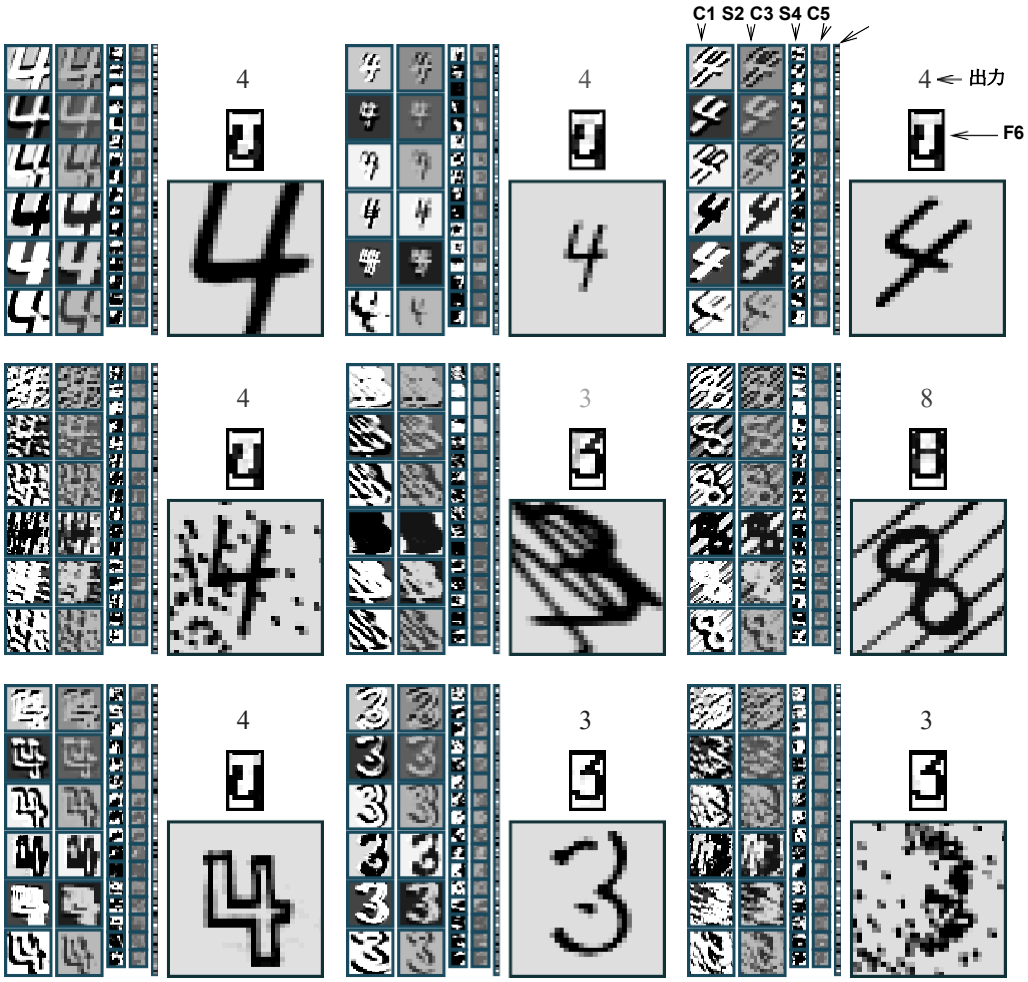


図13.LeNet-5で正しく認識された異常文字、歪んだ文字、ノイズのある文字の例。出力ラベルのグレーレベルはペナルティを表す（高いほど薄い）。

を fprop メソッドとして呼び出す。システムのすべての微分は，すべてのモジュールに対して bprop メソッドを，フォワードプロパティと逆の順序で呼び出すことによって計算することができる．状態変数には、フォワードパスで計算された状態のストレージに加え、バックワードパス1で計算された勾配を格納するスロットがあると仮定する。バックワードパスでは、システムのすべての状態変数とすべてのパラメータに関する損失Eの偏導関数が効果的に計算される。あるモジュールの前方関数と後方関数の間には、興味深い二元性があります。例えば、1前方方向の複数の変数の和は、後方方向では単純なファンアウト(複製)に変換される。逆に1、順方向のファンアウトは逆方向の和に変換される。本稿で紹介した結果を得るために使用したソフトウェア環境1「SN3.11」は、以上のような概念を用いている。SN3.11は、Lispのオブジェクト指向方言とC言語へのコンパイラをベースにした自作環境である。

微分が逆グラフの伝播によって計算できることは、直感的に理解しやすい。理論的にそれを正当化する最良の方法は、ラグランジュ関数 [21]1[22]を用いることである。同じ形式が

は、再帰的な接続を持つネットワークに手順を拡張するために使用されます。

B.特殊モジュール

ニューラルネットワークやその他多くの標準的なパターン認識技術は、勾配に基づく学習を用いたマルチモジュールシステムとして定式化することができる。一般的に使用されるモジュールには、行列の乗算やシグモイドモジュール1があり、これらの組み合わせにより、従来のニューラルネットワークを構築することができる。その他のモジュールとしては、畳み込み層1 サブサンプリング層1 RBF層1や「ソフトマックス」層などがあります[65]。損失関数もまた、単一の出力が損失の値を生成するモジュールとして表現される。一般的に使用されるモジュールは単純なbprop法である。一般に1 関数 F の bprop法は F のヤコビアンの乗算である。ここでは，よく使われる例をいくつか紹介する．ファンアウト(Y接続)のbpropメソッドはsum1であり，その逆もまた然りである．係数による乗算のbprop法は，同じ係数によるmultiplicationである．行列の乗算のbpropメソッドは、その行列のトランスポーズによる乗算です。定数との加算のbprop法は恒等式である。

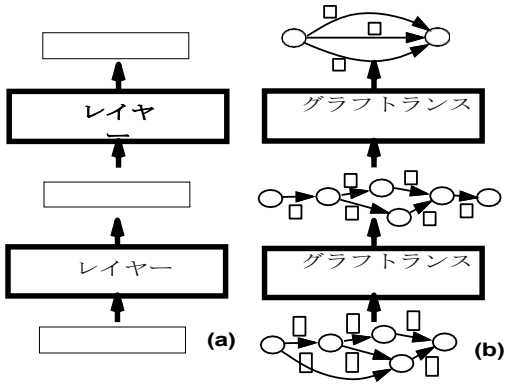


図15.従来のニューラルネットワークと、マルチモジュールシステムを組み合わせたコム
層間で固定サイズのベクトルを伝達する。多層グラフ変換ネットワークは、グラフを操作し、そのアークに数値が含まれるグラフを生成する学習可能なモジュールで構成されている。

興味深いことに1、ある種の非差別化モジュールは、マルチモジュールシステムに悪影響を与えることなく挿入することができる。その興味深い例として、マルチプレクサモジュールがある。このモジュールには、2つ（またはそれ以上）の通常入力1、スイッチング入力1、および出力1があります。このモジュールは、スイッチング入力1の（離散的な）値に応じて、その入力1の1つを選択し、出力にそれをコピーする。このモジュールは、スイッチング入力1に関しては区別できないが、通常の入力に関しては区別できる。したがって、このようなモジュールを含むシステムの全体的な機能は、スイッチング入力がパラメータに依存しない限り、そのパラメータに関して微分可能である。例えば1、スイッチング入力は外部入力でもよい。もう一つの興味深いケースは、minモジュールである。このモジュールは、2つ（またはそれ以上）の入力と1つの出力を持つ。このモジュールの出力は、入力の最小値である。このモジュールの関数は、測度ゼロの集合であるスイッチング面を除いて、どこでも微分可能である1。興味深いことに1、この関数は連続的かつ適度に規則的であり1、これは勾配学習アルゴリズム。

オブジェクト指向のマルチモジュール実装は、2次導関数のガウス・ニュートン近似を伝搬するbbpropメソッドを含むように簡単に拡張できる。これは、付録で示した2階微分バックプロパゲーション式22のモジュラーシステムに対する直接的な一般化につながる。

マルチプレクサモジュールは、セクション VIII1で詳しく説明した、システムのアーキテクチャが入力データに応じて動的に変化する、より一般的な状況1の特殊なケースである。マルチプレクサモジュールは、新しい入力パターンごとにシステムのアーキテクチャを動的に配線し直す（再構成する）ために使用することができます。

モジュール間1
は、すべて固定サイズのベクトルです。特に、可変長の入力（連続音声認識や手書き単語認識など）1や、数や性質が変化する物体や形状の間の関係を符号化する必要があるタスク（不変的知覚1 シーン解析1複合物認識）においては、データ表現における固定サイズベクトルの柔軟性の低さは多くのアプリケーションにおいて深刻な欠陥となる。重要な特殊ケースは、文字や単語の文字列の認識である。

より一般的には1、固定サイズベクターは、Lin-Spoonのようにベクトルやシンボルのシーケンスに対する確率分布を符号化する必要があるタスクには柔軟性に欠ける。

C.グラフトランスフォーマーネットワーク

マルチモジュールシステムは，大規模な訓練可能なシステムを構築するための非常に柔軟なツールである．しかし，前節までの説明では，パラメータ1のセットと状態情報の通信が暗黙のうちに前提となっていた．

ギー的な処理を行う。このような数列の分布は確率文法1 またはより一般的な有向グラフ1で表現され、各弧にベクトルが含まれる（確率文法はベクトルに確率と記号情報が含まれる特殊な場合である）。グラフの各経路は異なるベクトル列を表す。各弧に関連するデータの要素を確率分布のパラメータとして、あるいは単純にペナルティとして解釈することにより、系列に対する分布を表現することができる。シーケンス上の分布は、音声認識システムや手書き認識システムにおける言語知識のモデリングに特に便利である：各シーケンス1、すなわちグラフ1内の各パスは、入力の代替解釈を表す。連続する処理モジュールは、その解釈を徐々に洗練させていく。例えば、音声認識システムは、音響ベクトルの単一列から始まり、それを音素の格子（音素列に対する分布）1
に変換し、次に単語の格子（単語列に対する分布）1
に変換し、そして最適な解釈を表す単語の単一列に変換する。

大規模な手書き認識システム1
を構築する際に，システムを，1つまたは複数のグラフを入力として受け取り，出力としてグラフを生成するモジュールのネットワークとみなすことによって，より簡単かつ迅速に開発・設計できることを発見した．このようなモジュールをグラフ・トランスファー1
と呼び、完全なシステムをグラフ・トランスファー・ネットワーク1
またはGTNと呼んでいる。GTNのモジュールは、その状態や勾配を、数値情報（スカラーまたはベクトル）をアークに持つ有向グラフの形で伝達する[66]。

統計学的な観点1
から見ると、従来のネットワークの固定サイズの状態ベクトルは、状態空間における分布の平均を表すと見ることができる。VIII1 で述べた Space-Displacement Neural Net-works のような可変長のネットワークでは、状態は可変長の固定サイズのベクトル列である。これは固定長ベクトルの可変長シーケンスに対する確率分布の平均を表していると見ることができる。GTN1
では、状態はグラフ1
として表現され、構造化されたベクターの集まり（配列でもよい）に対する確率分布の混合として見ることができる （図15）。

次のいくつかのセクションの主要なポイントの一つは、勾配に基づく学習手順が、通信する単純なモジュールのネットワークに限定されないことを示すことである。

は、固定サイズのベクトル1
を通して計算されるが、GTN
に一般化することができる。グラフ変換器による勾配バックプロパゲーションは、出力グラフの数値情報に対して勾配をとり1、入力グラフの数値情報に対して勾配を計算する1、モジュール内部のパラメータに対して勾配を計算する。勾配学習は、入力グラフの数値データと関数のパラメータから出力グラフの数値データを生成する微分可能な関数であれば適用可能であり、入力グラフの数値データと関数のパラメータから出力グラフの数値データを生成する微分可能な関数であれば適用できる。

次に、一般に組み合わせ型と考えられている文書処理システム（およびその他の画像認識システム）1
の多くのモジュールで実装されている関数が、その内部パラメータおよび入力に対して実際に微分可能であり1
、したがってグローバルに学習可能なシステムの一部として使用できることを示すことである。

以下1
では、あえて確率論への言及を避けることにする。操作されるすべての量はペナルティ1　　またはコスト1
とみなされ、必要であれば指数関数的に正規化することで確率に変換することができる。

V.複数物体認識。ヒューリスティック・オーバーセグメンテーション

手書き認識における最も難しい問題の1つは、孤立した文字1だけでなく、郵便番号や小切手の金額、単語などの文字列1も認識することである。ほとんどの認識器は一度に1つの文字しか扱うことができないため、まず文字列を個々の文字画像に分割する必要があります。しかし、自然に書かれた文字列を確実に整形された文字に分割する画像解析技術を考案することは、ほとんど不可能である1。

最近の自動音声認識の歴史[28]1
[67]は、認識器を（単語または文レベルで）グローバルな基準を最適化することによって訓練することは、単に手で分割された音素または他のユニットで訓練するよりもはるかに好ましいことを思い出させてくれる。これは、認識器が個々の文字を認識するだけでなく、誤って分割された文字を拒否することもできるため、全体の単語誤差を最小にすることができるからである。

本節と次節では、単語や小切手などの文字列1
を読み取るための　　GTN
の単純な例について詳しく説明する。この方法は、個別にラベル付けされた文字画像で学習する従来のシステムでしばしば必要とされる、高価で信頼性の低いセグメンテーションの結果を手で確認する作業を回避することができる。

A.セグメンテーション・グラフ

単語のセグメンテーションと認識のための古典的な方法は、ヒューリスティック・オーバーセグメンテーション　　[68]1
[69]と呼ばれるものである。他のセグメンテーション手法と比較した場合の主な利点は次のとおりです。

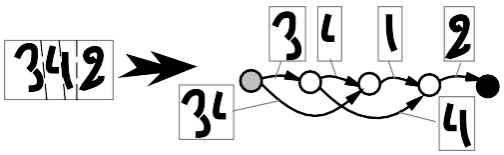


図16
ヒューリスティック・オーバー・セグメンテーションによるセグメンテーション・グラフの構築
ヒューリスティック・オーバー・セグメンテーションを用いたセグメンテーション・グラフの構築。

これは、多数の異なるセグメンテーションを考慮することで、セグメンテーションに関する難しい決定を避けることができるためである。このアイデアは、ヒューリスティックな画像処理技術を使用して、単語または文字列の候補カットを見つけ、次に認識器を使用して、それによって生成された変更されたセグメンテーションを採点することである。このプロセスは図16に示されている。まず、いくつかのカット候補が生成される。カットの良い候補位置は、垂直投影プロファイル1
の最小値、または単語の上部輪郭と下部輪郭の間の距離の最小値を見つけることによって見つけることができる。カット生成のヒューリスティックは，必要以上のカットを生成するように署名され　ており1，「正しい」カットのセットが含まれることが期待される．このような場合、「正しい」カットが含まれることを期待して、必要以上にカットを生成する。セグメンテーショングラフは、開始ノードと終了ノードを持つ有向非循環グラフ（DAG）である。各内部ノードは、セグメンテーション・アルゴリズムによって生成されたカット候補に関連付けられている。ソースノードとデスティネーションノード間の各円弧は、ソースノードに関連付けられたカットとデスティネーションノードに関連付けられたカットの間のすべてのインクを含む画像と関連付けられる。セグメンテーション担当者が、対応するカット間のインクがカナ文字になり得ると判断した場合、2つのノード間に円弧が作成される。通常、個々のインクが1つの円弧に関連付けられる。また、連続するインクのペアは、それらが異なる文字に属することを明確に示す大きなギャップ1
で区切られていない限り、含まれることになる。グラフを通る完全な経路は、各インクを一度だけ含む。各経路は、文字を形成するためにインクの断片を関連付ける異なる方法に対応する。

B.認識変換器とビタビ変換器

文字列を認識するための簡単なGTNを図17に示す。これは、認識変換器 T_{rec} とビタビ変換器 T_{vit} と呼ばれる2つのグラフ変換器からなる。認識変換器の目標は、解釈グラフまたは認識グラフ G_{int} と呼ばれるグラフ1
を生成することであり、これは入力のすべての可能な区分に対するすべての可能な相互予測を含むものである。G_{int}
の各パスは、入力の1つの特定のセグメンテーションの1つの可能な解釈を表す。ビタビ変換器の役割は、解釈グラフから最適な解釈を抽出することである。

認識変換器T_{rec} は、セグメンテーショングラフG_{seg} を入力1とし、各弧に関連する画像に単一文字用の認識器を適用する。

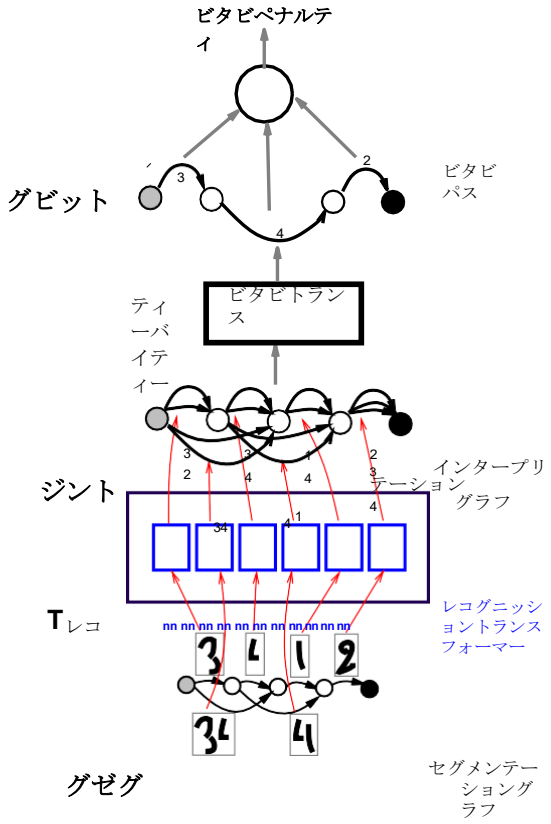
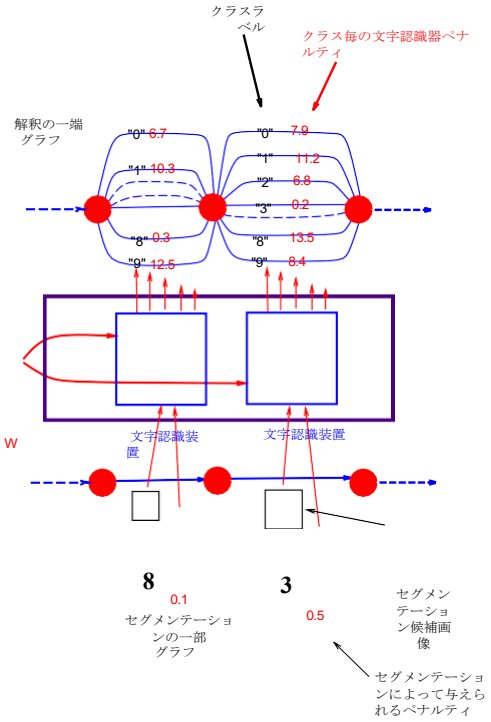


図17

GTNによる文字列の認識GTNによる文字列の認識。読みやすくするため、ペナルティの低い円弧のみを示している。



図

18.認識変換器は、セグメンテーションアークの各円弧を、解釈グラフの各文字クラスごとに、ペナルティーとラベルを付加した円弧の集合に洗練する。

を、セグメンテーション・グラフの中に置く。解釈グラフ G_{int} は、各円弧が同じノードからの円弧と同じノードへの円弧のセットに置き換えられることを除けば、セグメンテーション・グラフ 1 とほぼ同じ構造を持っている。この円弧のセット1 には、G_{seg} の対応する円弧に関連する画像の可能なクラスごとに1つの円弧がある。図 181 に示すように、各円弧には、クラスラベル1 と、認識器が生成したように画像がこのクラスに属するというペナルティが付加されている。セグメンテーション機能によって候補セグメント1 に対するペナルティが計算された場合、これらのペナルティは、文字認識機能1 によって計算されたペナルティと組み合わされて、解釈グラフのアークに対するペナルティが求められる。異なる性質のペナルティを組み合わせることは、非常に発見的であるように思われるが1 、GTN学習手順では、ペナルティを調整し、この組み合わせを利用することになる。解釈グラフの各パスは、入力された単語の解釈の可能性に対応する。あるセグメンテーションに対するある特定の解釈のペナルティは、解釈グラフの対応するパスに沿ったアークペナルティの合計で与えられる。ある解釈のペナルティをセグメンテーションとは無関係に計算するには、その解釈を持つすべてのパスのペナルティを結合する必要がある。並列パスのペナルティを結合するための適切な規則は、第VI-C節で述べる。

ビタビ変換器は、一つのパスを持つグラフ G_{vit} を生成する。このパスは、Interpretationグラフの累積ペナルティが最小のパスである。認識結果は、ビタビ変換器によって抽出されたグラフ G_{vit} に沿った円弧のラベルを読み取ることによって生成することができる。ビタビ変換器は、その名前を

有名なビタビアルゴリズム[70]1は、グラフ内の最短経路を効率的に求める動的計画法の原理を応用したものである。ソースノード s_i l とデスティネーションノード d_i を持つアーク $i1$ に関連するペナルティを c_i とする（2つのノード間に複数のアークが存在しうることに注意する）。解釈グラフ1では、アークもラベル l_i を持つ。ビタビアルゴリズムは以下のように進行する。各ノードnには、ビタビペナルティの累積値 v_n が設定される。これらの累積ペナルティは、解釈グラフ（有向かつ無サイクル）で定義される部分順序を満たす任意の順序で計算される。開始ノードは、累積ペナルティ v_{start} 0で初期化される。他のノードの累積ペナルティ v_n は、その親ノード1のv値から、宛先 d_i ngを持つ上流アーク U_n farc i を介して再帰的に計算される。

$$v_n \min(c_i + v_{s_i}). \quad (10)$$

さらに1、右辺を最小化する各ノードnのiの値は、最小化する進入弧を m_n lと記す。終点ノードに到達したとき、v_{end} で、総ペナルティが最小となる経路の総ペナルティを求める。このペナルティをビタビペナルティ1と呼び、この円弧とノードの列をビタビパスと呼ぶことにする。ノード $n_1 \ldots n_T$ とアーク $i_1 \ldots i_T$ _ 〃 持つビタビパスを得るには、これらのノードとアークを次のように辿る1 終点ノード n_T から始めて、開始ノードに達するまで最小の入るアーク：i_t m_{n_t} l と n_t s_{i_t} を使って再帰的に辿る。そして、ビタビパスの円弧からラベル列を読み取ることができる。