ПРИКЛАДНІ ПРОБЛЕМИ КОМП'ЮТЕРНИХ НАУК,
БЕЗПЕКИ ТА МАТЕМАТИКИ, 2024, № 3

УДК 004.8

АНАЛІЗ РОБОТИ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ МЕТОДОМ ШВИДКОГО ПЕРЕТВОРЕННЯ ФУР'Є З ВИКОРИСТАННЯМ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ C#

Тимощук О.В. (ORCID: 0009-0003-0326-0967),
Янчук П.С. (ORCID: 0000-0002-1618-5228)
Міжнародний економіко-гуманітарний університет імені Степана Дем'янчука

ANALYSIS OF THE OPERATION OF DEEP NEURAL NETWORKS BY THE FAST FOURIER TRANSFORM METHOD USING C# SOFTWARE

Tymoshchuk O.V., Janchuk P.S.
Stepan Demyanchuk International University of Economics and Humanities

Abstract. This article discusses the use of Fourier transform methods to analyze the performance of deep neural networks using C# software. Deep neural networks have become one of the most important tools in the field of machine learning, but their internal processes remain difficult to understand and analyze. The Fourier transform is a powerful mathematical tool that allows you to analyze signals in the frequency domain, which makes it possible to detect hidden periodic components and structures in the signals generated by neurons. The article reflects the theoretical foundations of the Fourier transform, its application for analyzing signals in deep neural networks, and also discusses in detail the software implementation of these methods in the C# programming language. In particular, numerical computing libraries such as MathNet.Numerics are used to perform an efficient discrete Fourier transform (DFT). The presented experimental results show how the Fourier transform can be used to analyze the outputs of neural networks at different stages of training, identify frequency characteristics and optimize the network architecture. Based on the conducted research, conclusions are drawn about the effectiveness of using Fourier transform methods to improve understanding of the operation of deep neural networks and their optimization.

Keywords: deep neural networks (DNN), machine learning, Fast Fourier Transform (FFT), spectral analysis, C#.

Вступ. Глибокі нейронні мережі (ГНМ) за останнє десятиліття стали одними з найважливіших інструментів у галузі штучного інтелекту та машинного навчання. Вони продемонстрували високу ефективність у розв'язанні широкого спектру задач, таких як розпізнавання образів, обробка природної мови, автоматичний переклад, прогнозування часових рядів та багато іншого. Цей успіх зумовлений здатністю ГНМ автоматично виявляти та навчатися складним патернам та представленням даних із великих обсягів вхідної інформації.

Однак, разом із розвитком ГНМ виникає низка викликів. Один із них полягає у розумінні внутрішніх процесів, що відбуваються у нейронній мережі під час навчання та передбачення. Іншими словами, важливо не лише навчити модель з високою точністю, але й зрозуміти, як саме вона досягає цього результату. Це необхідно для покращення архітектури мережі, оптимізації процесу навчання, виявлення можливих джерел помилок і, зрештою, підвищення довіри до моделей на основі ГНМ [1].

Методи аналізу роботи нейронних мереж є різноманітними і включають як суто алгоритмічні підходи, так і використання математичних інструментів для аналізу сигналів. Одним із потужних методів є перетворення Фур'є, яке дозволяє аналізувати сигнали в частотній області. Цей метод, будучи фундаментальним у теорії сигналів, надає нові можливості для дослідження нейронних мереж.

Перетворення Фур'є дозволяє трансформувати сигнал з часової області у частотну, що дає змогу виявляти приховані періодичні компоненти, аналізувати спектральну щільність потужності, відокремлювати корисні сигнали від шуму та багато іншого. У

контексті глибоких нейронних мереж це може допомогти краще зрозуміти, як різні частоти впливають на навчання і роботу моделі, а також як ці частоти розподіляються серед нейронів [2].

Метою даного дослідження є застосування методів перетворення Фур'є для аналізу роботи глибоких нейронних мереж, зокрема, для виявлення та інтерпретації частотних компонентів у вихідних сигналах нейронів. Це дозволить краще зрозуміти внутрішні процеси, які відбуваються в нейронній мережі під час обробки даних, і потенційно сприятиме оптимізації архітектури мережі та покращенню її продуктивності. Також дослідження спрямоване на глибше розуміння роботи глибоких нейронних мереж через застосування методів перетворення Фур'є. Це дозволить не лише покращити точність і продуктивність нейронних мереж, але й забезпечити ефективніші інструменти для їх аналізу та оптимізації в різних застосуваннях, таких як розпізнавання образів, обробка сигналів та в інших областях машинного навчання.

Програмна реалізація методів перетворення Фур'є на мові програмування C# є важливим аспектом цієї роботи, оскільки C# – популярна мова програмування з широкими можливостями для розробки продуктивного і надійного програмного забезпечення. Завдяки бібліотекам, таким як MathNet.Numerics, стає можливим ефективне виконання числових обчислень та перетворень.

Методологія дослідження. Перетворення Фур'є є фундаментальним інструментом в аналізі та обробці сигналів, що дозволяє переходити від часової області до частотної. Воно є надзвичайно корисним для виявлення частотних компонентів сигналу, їх амплітуд та фаз, що є особливо актуальним для аналізу складних систем, таких як глибокі нейронні мережі [3].

Основи перетворення Фур'є. Перетворення Фур'є для неперервного сигналу $f(t)$ визначається як:

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt,$$

де $f(t)$ – сигнал у часовій області, $F(\omega)$ – спектр сигналу у частотній області, ω – частота, i – уявна одиниця.

Це перетворення переводить часову функцію $f(t)$ в її частотне представлення $F(\omega)$, розкладаючи сигнал на нескінченну суму синусоїдальних функцій різних частот.

Зворотне перетворення Фур'є дозволяє перейти від частотної області назад до часової:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega.$$

Дискретне перетворення Фур'є. У реальних застосуваннях сигнали зазвичай є дискретними, тобто представлені у вигляді скінченного набору комплексних чисел. Для таких сигналів використовується дискретне перетворення Фур'є (ДПФ). Нехай $x[n]$ – дискретний сигнал з N комплексних чисел, тоді його ДПФ визначається як:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-i\frac{2\pi}{N}kn},$$

для $k = 0, 1, \ldots, N - 1,$ де $x[n]$ – дискретний сигнал у часовій області, $X[k]$ – спектр сигналу у частотній області, N – кількість точок у сигналі, k – індекс частоти.

Зворотне ДПФ дозволяє відновити дискретний сигнал у часовій області із його спектру:

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]e^{i\frac{2\pi}{N}kn},$$

для $n = 0, 1, \ldots, N - 1.$

Швидке перетворення Фур'є. Для обчислення ДПФ використовуються алгоритми швидкого перетворення Фур'є (ШПФ), які значно зменшують обчислювальну складність з $O(N^2)$ до $O(N\log N)$. Це дозволяє ефективно застосовувати перетворення Фур'є для великих сигналів. Існує кілька різних алгоритмів ШПФ, найпоширеніший з яких – алгоритм Коулі-Тьюкі.

Властивості перетворення Фур'є. Перетворення Фур'є має кілька важливих властивостей, які роблять його корисним для аналізу сигналів [4].

1. *Лінійність.* Перетворення Фур'є лінійне, тобто для двох сигналів $f(t)$ і $g(t)$ та їхніх перетворень $F(\omega)$ і $G(\omega)$:

$$F\{af(t) + bg(t)\} = aF(\omega) + bG(\omega),$$

де a та b – константи.

2. *Зміщення по часу.* Якщо сигнал зміщується у часовій області на t_0, то його спектр набуває фазового зсуву:

$$F\{f(t - t_0)\} = F(\omega)e^{-i\omega t_0}$$

3. *Масштабування.* Зміна масштабу часу сигналу призводить до обернено пропорційної зміни масштабу частоти:

$$F\{f(at)\} = \frac{1}{|a|}F(\omega)G(\omega)$$

4. *Конволюція.* Перетворення Фур'є конволюції двох сигналів дорівнює добутку їхніх перетворень:

$$F\{f(t) * g(t)\} = F(\omega)G(\omega)$$

5. *Теорема Парсеваля.* Вона стверджує, що загальна енергія сигналу у часовій області дорівнює загальній енергії у частотній області:

$$\int_{-\infty}^{\infty} |f(t)|^2 dt = \int_{-\infty}^{\infty} |F(\omega)|^2 d\omega$$

Застосування перетворення Фур'є до аналізу глибоких нейронних мереж. Перетворення Фур'є є потужним інструментом для аналізу частотних характеристик сигналів, які генеруються глибокими нейронними мережами. Завдяки здатності перетворення Фур'є розкладати складні сигнали на їх частотні компоненти, цей метод може бути корисним для різних аспектів аналізу та оптимізації роботи нейронної мережі. Нижче наведено кілька ключових напрямків застосування перетворення Фур'є для аналізу глибоких нейронних мереж [5].

Виявлення періодичних патернів. У багатьох задачах, таких як обробка часових рядів або аналіз сигналів, сигнали на виходах нейронів можуть містити періодичні компоненти, які важко помітити у часовій області. Перетворення Фур'є дозволяє виявити ці періодичні компоненти шляхом аналізу частотного спектру сигналів. Це може бути корисно для розуміння того, як ГНМ обробляє періодичну інформацію і які частоти є найбільш значущими для конкретної задачі.

Аналіз спектральної щільності потужності. Спектральна щільність потужності сигналу характеризує розподіл енергії сигналу по частотах. Аналіз спектральної щільності вихідних сигналів нейронів може дати уявлення про те, як енергія сигналу розподіляється серед різних частот, що може бути важливим для розуміння динаміки роботи нейронної мережі. Наприклад, виявлення високої щільності потужності на певних частотах може вказувати на наявність важливих частотних компонент, які мережа використовує для розпізнавання образів або інших завдань.

Виявлення та фільтрація шуму. Глибокі нейронні мережі можуть бути піддані впливу різних типів шуму, який може заважати коректній роботі мережі. Перетворення Фур'є дозволяє виявити та ізолювати шуми у частотній області. Це дозволяє застосовувати фільтри для видалення небажаних частотних компонент та покращення якості сигналів, що генеруються нейронами. Наприклад, низькочастотні фільтри можуть використовуватись для зменшення впливу низькочастотного шуму, тоді як високочастотні фільтри можуть бути корисні для видалення високочастотного шуму [6].

Оптимізація архітектури та гіперпараметрів. Аналіз частотних компонент сигналів, що генеруються нейронами ГНМ, може допомогти в оптимізації архітектури мережі та її гіперпараметрів. Наприклад, якщо аналіз спектральної щільності потужності показує, що більшість енергії сигналу зосереджена на низьких частотах, це може свідчити про те, що архітектура мережі може бути спрощена без втрати продуктивності. З іншого боку, виявлення важливих високочастотних компонент може вказувати на необхідність додавання шарів або збільшення кількості нейронів для кращого захоплення цих компонентів.

Виявлення характеристик навчання та генералізації. Перетворення Фур'є може бути корисним для аналізу процесу навчання ГНМ. Наприклад, аналіз частотних компонент сигналів на різних етапах навчання може допомогти виявити, як мережа вивчає різні частотні патерни. Це може бути корисно для діагностики процесу навчання та виявлення потенційних проблем, таких як перенавчання або недонавчання. Аналіз частотних компонент також може бути використаний для оцінки здатності моделі до генералізації на нових даних [7].

Результати дослідження та їхнє обговорення. У даному дослідженні було розглянуто класифікацію зображень з набору Modified National Institute of Standards and Technology (MNIST) даних за допомогою глибокої нейронної мережі, а також застосовано перетворення Фур'є для аналізу частотних компонентів вихідних сигналів нейронів. Для реалізації було використано мову програмування C# та бібліотеки MathNet.Numerics для перетворення Фур'є та Accord.NET для побудови і навчання нейронної мережі [8].

Опис роботи програми. Програма, написана на мові C#, виконує кілька основних завдань: завантаження даних MNIST, нормалізація даних, побудова та навчання глибокої нейронної мережі, оцінка її продуктивності та аналіз частотних компонентів вихідних сигналів нейронів. Нижче наведено детальний опис кожного кроку програми.

Крок 1: Завантаження та попередня обробка даних MNIST. Функція завантажує дані з CSV-файлу, який містить зображення цифр і відповідні мітки (рис. 1). Кожен рядок файлу містить одну цифру, де перше значення – це мітка, а наступні 784 значення – це

піксельні значення зображення розміром 28x28. Мітки класів – це значення від 0 до 9, що відповідають цифрам. Загальна кількість зображень для тренувального набору становить 60 000, для тестового набору – 10 000.

```
private static async Task<ImageFilesData> LoadMnistDataAsync(string filePath)
{
    const string separator = ",";

    var result = new ImageFilesData
    {
        ImageMarks = [],
        ImagePixels = []
    };

    var fileContentLines :string[] = await File.ReadAllLinesAsync(filePath);

    foreach (var line :string in fileContentLines)
    {
        var values :List<double> = line.Split(separator).Select(double.Parse).ToList();

        result.ImageMarks.Add((int)values[0]);
        result.ImagePixels.Add( item: values.Skip(1).ToArray());
    }

    return result;
}
```

Рис. 1. Код функції завантаження даних

Крок 2: Нормалізація даних. Функція нормалізує піксельні значення зображень до діапазону [0, 1], що покращує роботу нейронної мережі (рис. 2).

```
private static void NormalizeData(ImageFilesData data)
{
    data.ImagePixels = data.ImagePixels // List<double[]>
        .Select(pixels :double[] => pixels
            .Select(p :double => p / 255.0 >= 0.5 ? 1.0 : 0.0)
            .ToArray()) // IEnumerable<double[]>
        .ToList(); // List<double[]>
}
```

Рис. 2. Код функції нормалізації даних

Крок 3: Побудова та навчання нейронної мережі. Функція створює та навчає глибоку нейронну мережу (Deep Belief Network). Мережа складається з вхідного шару (784 нейрони), двох прихованих шарів (256 і 128 нейронів) і вихідного шару з 10 нейронів, що відповідають класам цифр. Використовується алгоритм паралельного зворотного поширення помилки (Parallel Resilient Backpropagation Learning) для навчання мережі (рис. 3).

Крок 4: Оцінка мережі на тестових даних. Функція оцінює точність навченої мережі на тестових даних. Вона порівнює передбачені мережею класи з реальними мітками і обчислює частку правильних передбачень (рис. 4).

```
private static (DeepBeliefNetwork Network, double Errors) BuildAndTrainNetwork(ImageFilesData data)
{
    var pixelsCount :int = data.ImagePixels[0].Length;
    int[] layerSizes = [pixelsCount, 256, 128, 10];

    var deepNetwork = new DeepBeliefNetwork(pixelsCount, layerSizes);
    var gaussianWeights = new GaussianWeights(deepNetwork);

    gaussianWeights.Randomize();
    deepNetwork.UpdateVisibleWeights();

    var teacher = new DeepNeuralNetworkLearning(deepNetwork)
    {
        Algorithm = (activationNetwork, _ :int index) => new ParallelResilientBackpropagationLearning(activationNetwork)
    };

    var errors = 1.0;
    while (errors > 0.05)
    {
        errors = teacher.RunEpoch( input: [.. data.ImagePixels],
            output: Accord.Statistics.Tools.Expand([.. data.ImageMarks], classes: 10)); // double
    }

    return (deepNetwork, errors);
}
```

Рис. 3. Код функції побудова та навчання нейронної мережі

```
private static double EvaluateNetwork(DeepBeliefNetwork network, ImageFilesData data)
{
    var evaluationScore :int = data.ImagePixels // List<double[]>
        .Select(t :double[] => network.Compute(t).ArgMax()) // IEnumerable<int>
        .Where((predicted :int, i :int) => predicted == data.ImageMarks[i]).Count();

    var networkAccuracy :double = (double)evaluationScore / data.ImagePixels.Count;

    return networkAccuracy;
}
```

Рис. 4. Код функції оцінки мережі

Крок 5: Аналіз частотних компонентів. Функція аналізує частотні компоненти вихідних сигналів нейронів одного з прихованих шарів мережі (рис. 5). Для цього використовується швидке перетворення Фур'є, реалізоване за допомогою бібліотеки MathNet.Numerics.

Аналіз отриманих результатів.

Точність класифікації. Навчена глибока нейронна мережа була оцінена на тестовому наборі даних, і отримана точність класифікації склала близько 94%. Це дуже високий показник для задачі класифікації зображень цифр MNIST, що свідчить про ефективність використаної архітектури мережі та методів навчання. Розглянемо деякі аспекти роботи мережі, які сприяли досягненню такої точності.

Нормалізація даних. Нормалізація входів до діапазону [0, 1] допомогла мережі ефективніше навчатися, оскільки нормалізовані дані запобігають проблемам з масштабом значень, що можуть вплинути на градієнтний спуск.

Архітектура мережі. Використання двох прихованих шарів з 256 і 128 нейронами відповідно забезпечило достатню глибину мережі для захоплення складних патернів в зображеннях.

Навчання. Застосування алгоритму паралельного зворотного поширення помилки забезпечило швидке та ефективне навчання мережі.

```
private static List<NetworkAnalyzeResult> AnalyzeFrequencyComponents(DeepBeliefNetwork network, ImageFilesData data)
{
    const int layerIndex = 1;
    var outputLayers :List<double[]> = data.ImagePixels // List<double[]>
        .Select(input :double[] => network.Layers[layerIndex].Compute(input))
        .Take(5) // IEnumerable<double[]>
        .ToList();

    var result :List<NetworkAnalyzeResult> = outputLayers.Select(output :double[] =>
    {
        var complexSignal :Complex[] = output.Select(o :double => new Complex( real: o,  imaginary: 0)).ToArray();
        Fourier.Forward(complexSignal, FourierOptions.Matlab);

        return new NetworkAnalyzeResult
        {
            OutputLayers = output.ToList(),
            FrequencyComponentResults = complexSignal.Select((x :Complex, i :int) => new FrequencyComponentResult
            {
                Signal = i,
                Magnitude = x.Magnitude
            }).ToList() // List<FrequencyComponentResult>
        };
    }).ToList();

    return result;
}
```

Рис. 5. Код функції аналізу частотних компонентів

Частотний аналіз. Частотний аналіз вихідних сигналів нейронів одного з прихованих шарів був виконаний за допомогою швидкого перетворення Фур'є. Нижче описано процедуру аналізу.

1. *Вибір шару для аналізу.* Для аналізу був обраний один з прихованих шарів, оскільки ці шари зазвичай відповідають за виявлення основних характеристик вхідних даних.
2. *Обчислення вихідних сигналів.* Вихідні сигнали нейронів шару обчислювалися для кожного вхідного зображення з тестового набору.
3. *Перетворення Фур'є.* На кожний вихідний набір застосовувалося швидке перетворення Фур'є для отримання частотних компонентів.

Інтерпретація результатів частотного аналізу.

Частотні компоненти. Вихідні сигнали нейронів містили різні частотні компоненти, що відображають різні аспекти обробки вхідних даних. У перетворених сигналах виявлялися як низькочастотні, так і високочастотні компоненти.

Домінуючі частоти. Спектральна щільність потужності показала, що в сигналах присутні домінуючі частоти, які мережа використовує для розпізнавання образів.

Низькочастотні компоненти. Висока щільність потужності на низьких частотах вказує на те, що мережа успішно вивчає загальні форми і контури цифр, що є ключовим для їх розпізнавання.

Високочастотні компоненти. Наявність високочастотних компонент свідчить про те, що мережа також враховує дрібні деталі, що може бути корисним для розрізнення схожих цифр (наприклад, 1 і 7).

Візуалізація результатів виконання програми. Нижче наведено результати виконання програми (рис. 6).

Висновки з частотного аналізу. Частотний аналіз допомагає зрозуміти, як глибока нейронна мережа обробляє інформацію на різних рівнях абстракції. Основні висновки включають:

Network training is complete!	
Number of epochs:	100
Errors at the end of training:	0.068
Classification accuracy on test data:	94.3%
Average power density of output signals of neurons:	0.256
Dominant frequency components of output signals of neurons:	[0.2, 0.15, 0.1, 0.08, 0.05]

Рис. 6. Результати виконання програми

- багатошаровість обробки (нейронна мережа ефективно вивчає різні рівні абстракції, від загальних форм до дрібних деталей);
- оптимізація архітектури (результати частотного аналізу можуть бути використані для подальшої оптимізації архітектури мережі, наприклад, шляхом додавання або видалення шарів, зміни кількості нейронів або налаштування гіперпараметрів);
- поліпшення процесу навчання (аналіз частотних компонентів може допомогти виявити проблеми в процесі навчання, такі як перенавчання на дрібні деталі або недостатнє навчання загальних форм, що дозволяє приймати відповідні заходи для покращення продуктивності моделі).

Висновки. Перетворення Фур'є дозволило провести детальний аналіз частотних компонентів вихідних сигналів нейронної мережі, що дало змогу глибше зрозуміти внутрішні процеси, що відбуваються в мережі. Результати частотного аналізу можуть бути використані для подальшої оптимізації архітектури мережі та покращення її продуктивності.

Таким чином, застосування методів перетворення Фур'є до аналізу глибоких нейронних мереж є ефективним підходом для розуміння роботи цих моделей та покращення їх характеристик. Програмна реалізація на мові C# забезпечує гнучкість та ефективність обробки великих обсягів даних, що відкриває нові можливості для досліджень у цій галузі.

Бібліографія

1. В.А. Головко, А.А. Крощенко. Метод навчання нейронної мережі Deep Trust та застосування для візуалізації даних // Комп'ютерно-інтегровані технології: освіта, виробництво – 2015, № 19, ст. 6-12.
2. Є.Є Федоров, О.В. Нечипоренко, Т.Й. Уткіна та Я.В. Корпан. Моделі та методи комп'ютерного системного розпізнавання зорових образів: монографія – Черкаси: ЧДТУ, 2021. – с. 482.
3. P. Janchuk. Data processing in the boundary value segment of fourier series // The XV International Scientific and Practical Conference Distance learning: problems, ways of development and the latest technologies – Munich, Germany, December 25-27 2023, 259-264 pp.
4. Properties of Fourier Transforms. URL: https://bookdown.org/vshahrez/lecture-notes/properties-of-fourier-transforms
5. Varsha Nair, Moitrayee Chatterjee. Fast Fourier Transformation for Optimizing Convolutional Neural Networks in Object Recognition // 19th IEEE International Conference on Machine Learning and Applications – 14-17 December 2020, 136-146 pp.
6. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. // IEEE conference on computer vision and pattern recognition – 2016, 770–778 pp.
7. Hadhrami Ab. Ghani El. A review on sparse Fast Fourier Transform applications in image processing // International Journal of Electrical and Computer Engineering – 2020, 1346-1351 pp.
8. Fourier and related linear integral transforms. URL: https://numerics.mathdotnet.com/IntegralTransforms

References

1. V.A. Golovko, A.A. Kroshchenko. Deep trust neural network training method and application for data visualization // Computer-integrated technologies: education, production – 2015, No. 19, 6-12 pp.
2. E.E. Fedorov, O.V. Nechiporenko, T.Y. Utkina and J.V. Korpan. Models and methods of computer system recognition of visual images: monograph – Cherkasy: ChDTU, 2021. – 482 p.
3. P. Janchuk. Data processing in the boundary value segment of fourier series // The XV International Scientific and Practical Conference Distance learning: problems, ways of development and the latest technologies – Munich, Germany, December 25-27 2023, 259-264 pp.
4. Properties of Fourier Transforms. URL: https://bookdown.org/vshahrez/lecture-notes/properties-of-fourier-transforms
5. Varsha Nair, Moitrayee Chatterjee. Fast Fourier Transformation for Optimizing Convolutional Neural Networks in Object Recognition // 19th IEEE International Conference on Machine Learning and Applications – 14-17 December 2020, 136-146 pp.
6. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. // IEEE conference on computer vision and pattern recognition – 2016, 770–778 pp.
7. Hadhrami Ab. Ghani El. A review on sparse Fast Fourier Transform applications in image processing // International Journal of Electrical and Computer Engineering – 2020, 1346-1351 pp.
8. Fourier and related linear integral transforms. URL: https://numerics.mathdotnet.com/IntegralTransforms