Contents

1 Introduction

1.1 Intro to printed electronics

Printed electronics refers to very thin electronic devices and circuits that are produced by the application of inks with desired electric properties to various substrates. They can be manufactured in volume for a much lower cost compared to other electronics with methods common in the printing industry. This makes them particularly well-suited for applications where the benefits of electronic functionality alone do not outweigh the associated expenses. Additionally they can offer flexible form factors and the ability of large area coverage. Another benefit that may come from their spread is to lessen the impact of e-waste, since printed electronics can be much less toxic for the environment and more easily recyclable than the rest, or even biodegradable. They cannot compete with silicon electronics in performance due to the large resistance of conductive inks, the lack of support for high frequency and the high variability in manufacturing. While the ability to cover large areas is sometimes desirable, a lot of applications demand miniaturization that they cannot offer. A variety of active and passive devices, including transistors, resistors, capacitors, sensors, harvesters and antennas can be implemented with them. They are thought to be an emerging market with considerable potential to broaden the role of computation in everyday living. They can help the pervasiveness of the Internet-of-Things reach far deeper, and thus synergize well with other advances in the sector. A recent report by IDTechEx[17] forecasts the global market for printed flexible electronics, excluding OLEDs, to reach 12 billion dollars by 2033.

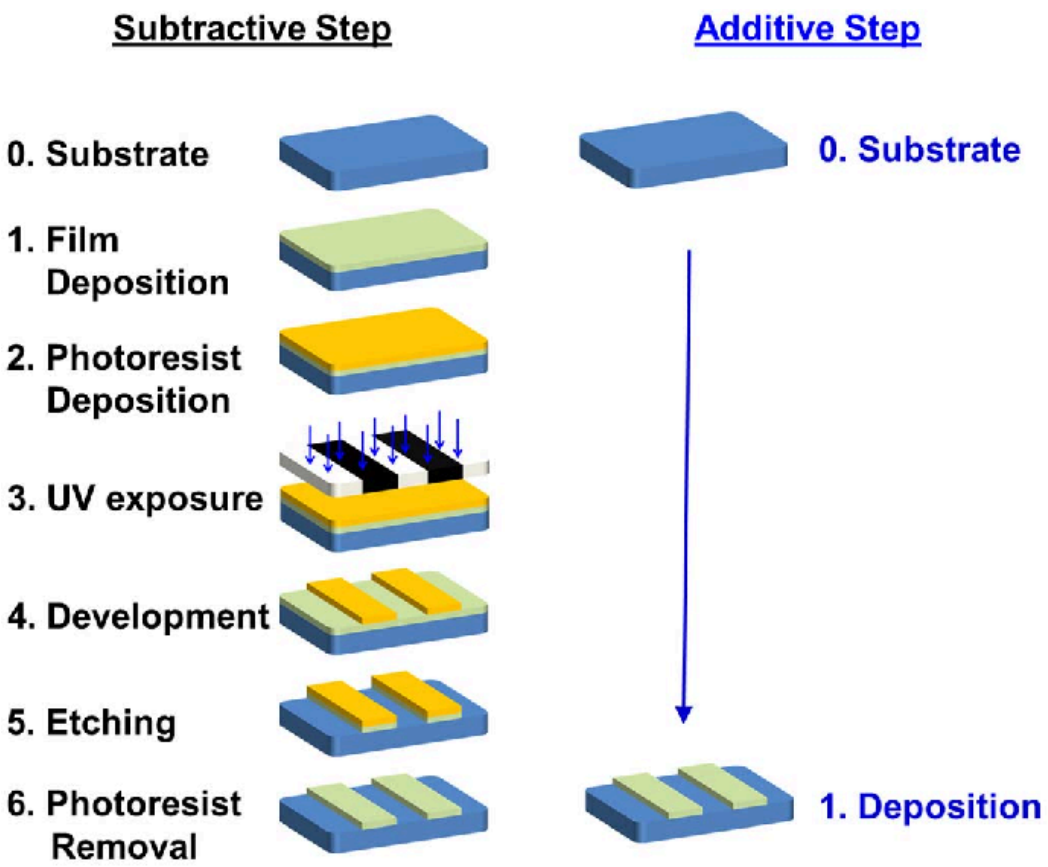

Figure 1: Source: https://doi.org/10.1109/ISCAS.2017.8050614

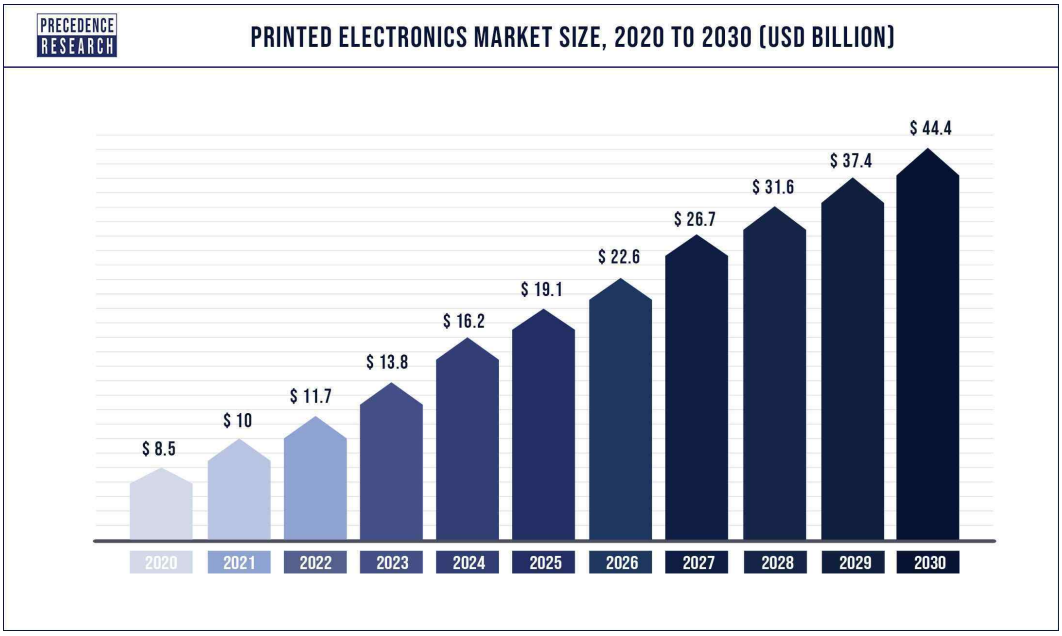

Figure 2: Source: Precedence Research

2 Related works in machine learning for printed circuits

Since the printed computing technology reached the point where Machine Learning models could be supported for inference, work has been made in bringing them to fruition. Tahoori et al[29] demonstrates an analog two input neuron, and shows how it could be expanded to fully printed analog neural networks with MAC and activation operations. Douthwaite et al[13] Uses time domain encoding of signals, representing magnitude as pulse width and encoding weights with current mirrors. Accumulation is done by linearly charging a capacitor with the mirrored pulses. Gkoupidenis et al [23] mimick biologically inspired synaptic functions with electrolyte-gated transistors and show how they could be used for a single layer perceptron. Ozer et al [25] envision what an automatic process for creating bespoke processors for a variety of ML architectures in printed electronics could look like, but don't go beyond the vision stage. Bleier et al [9] present a printed microprocessor with an instruction set customised to the program at hand. Weller et al [28] leverage stochastic computing to reduce the requirements of mixed analog-digital neural networks but with heavy accuracy cost.

Mubarik et al [24] evaluate small machine learning architectures (decision trees, random forests and support vector machines) in digital, lookup table based and analog architectures in bespoke printed circuits. They also consider MLPs, but decide they are too costly to evaluate. The main results are about the decision trees(DTs), where they examine the demands of printed implementations for depths of 1 to 8, in both conventional and bespoke circuits. They show that bespoke circuits, that are uniquely suited to printed electronics due to low non-recurring engineering(NRE) and manufacturing costs, can be implemented with about two orders of magnitude lower requirements than circuits that can support a wider range of DTs and not just one. This thesis is directly inspired by this work in using bespoke design to lower the demands of model implementations and mainly applies their insights to the domain of BNNs.

Armeniakos et al [5] expand to more demanding SVMs and Multi Layer Perceptrons. In order to enable their implementation, they leverage approximate computing in two ways. First they notice that there is high variance in the area demands of a constant multiplier based on the coefficient it multiplies by. For example, multiplying by a power of two takes no hardware at all since it is a constant shift. They approximate weight coefficients of the MLPs and SVMs to take advantage of this observation. Secondly, they apply post-synthesis pruning at the gate level on the netlist of the designs. They target gates that have close to constant outputs and only

influence less significant bits of the results and replace them with the constant value they mostly output. Together these approximations result in area and power reductions of about a factor of 2 in most cases. This work is the direct inspiration of this thesis, where the weight coefficients are set in the training phase to be exclusively values that don't require multipliers to be implemented, as is the case in BNNs. The results achieved here are thus compared with the ones from this work as a baseline. This comparison is provided in the Results section.

In the follow up paper [22] they additionally apply neural minimization techniques such as quantization, pruning and weight clustering and combine them utilising genetic algorithms to reduce area requirements by up to 8x.

In [3], in addition to the aforementioned hardware-friendly coefficients and netlist pruning, voltage overscaling(VOS) is applied to further reduce the power demands of classifier circuits. A genetic algorithm is then applied to minimize area and maximise accuracy for a given power constraint. This enables many designs to be powered by printed batteries sacrificing less that 1% in accuracy.

[4] retrains MLPs with a scoring function that takes the hardware cost associated with multiplying with each weight's coefficient into account. Coefficients are sorted into clusters based on their hardware cost and retraining allows increasingly more expensive values to be used for weights until the accuracy threshold is met. Additionally the products are summed using approximate addition by discarding the least significant bits of products that contribute less to the MAC's result. Together these improvements lead to 6x area and power savings for 1% accuracy loss and 20x for 5%. Because often their networks use only powers of 2 as weights and thus no hardware is used to perform multiplication, this edge of using BNNs is not present here. However different neurons use different weights for the same input, leading in less intermediate sums being shareable across neurons. This is an edge BNNs can exploit, albeit paying a price in representational capabilities.

[7] extends the idea of hardware-friendly coefficients to the threshold values of comparators in decision trees. Beyond the threshold value the precision of the comparison can also be configured at a per-comparator basis in order increase efficiency. They deploy a genetic algorithm to find optimal configurations of hardware-friendly thresholds close to the original values and reduced comparison precisions without sacrificing more than 1% accuracy. As a result area and power are reduced 3-4x. This leads some of the smaller designs they examine to sub-cm^2 area and sub-mW power draw.

Iordanou et al [18] have an interesting approach in which they use graph-based genetic programming to search the space of boolean logic expressions for ones that

predict the class of tabular data with high accuracy and transpiling those logic gates into a netlist. The result is a sea of logic gates, unlike the structured circuits of other approaches. Needless to say this is removed from the paradigm of traditional ML architectures this work is placed in.

3 Introduction to ubiquitous computing and thesis statement

Technology in general and more specifically computation plays an ever increasing part in our lives and there are no signs of the trend slowing down any time soon. There still however exists a relatively rigid real world - computational domain gap, meaning most of of our interactions of the world around us don't involve any computation taking place. It is not hard to imagine countless examples where computational elements would add value to everyday activities such as grocery shopping or reduce required labour in production processes such as manufacturing if those elements had close to zero cost and greater embedability associated with them. Although almost everyone in developed countries carries and interacts with powerful computers everywhere they happen to be, the form of interaction cannot easily adapt to the surrounding context they are in. One cannot simply ask the bananas they got if they are ripe enough, call out to their keys to find where they left them, check with their shoes on how many more steps they got in them. Furthermore it is clear that uncountable processes are horribly unoptimised compared to what could be achieved if a continuous stream of detailed information from each of of it's constituents and access to fine grained control over the minutia of them was in place. Think for example a farm where every individual fruit on any tree has it's growth progress tracked. Essentially taking the ideas of the Internet-of-Things(IoT) and pushing them to their logical limit, ubiquitous computing is an aspirational ideal to a future where every product is a smart device, every observable anyone would reasonably care for is accessible. Self driving cars will be able to safely navigate without access to vision by querying the positions of nearby devices, since whatever is not a device directly at least has one or more attached.

Printed electronics are positioned to play a major role in at the very least the early stages of such a transformation. Printing is currently the only manufacturing method that can provide sub-cent computational elements, and cost is the greatest bottleneck to how pervasive they can become. Additionally the non-toxicity is crucial to make adding them to fast moving consumer goods that are disposable at these scales. The flexibility also helps with embedding more easily. Even relatively "modest" compared to the complete vision applications that we can expect to come eventually, such as RFID tags replacing barcodes and enabling stores to track every individual item of stock or printed food quality sensors making best-before dates obsolete have great potential to disrupt a wide range of industries.

Machine learning can accelerate the process by many orders of magnitude compared to how long it would take for specialised people in multiple fields to design a com-

putational model to interpret and process the sensor data. In many cases the upfront engineering cost would be enough to stop the adoption of the paradigm altogether. If all it takes is for some sensor data to be collected and labeled that can easily be handled by any employee. We also depend on autoML being good enough for most of these small scale data applications, since otherwise we would just run into the same bottleneck with needing a data scientist for every little thing. Methods to lighten the resources demanded of the printed system that implements the model, such as quantization and binarization can clearly expand the scope of how complex the classification supported can be.

The concept of this thesis is taking place in such a scenario. I have insured that the entire process from dataset to netlist that can be passed to the printer requires no manual intervention. Anyone can pass their sensor data in one end and receive measurements for the model accuracy, circuit area and power requirements on the other, without special knowledge on any domain being required of them. This is specifically done utilising bespoke implementations of binary neural networks, in order to evaluate their efficacy in providing a backbone for this process.

Imagine if you will the scenario of a coffee shop owner. They decide they would like the glasses they serve their coffee in to indicate the amount of sugar or other sweeteners used in the contained beverage. This would prevent people from grabbing the wrong coffee from the table because they all look indistinguishable. After searching on an online repository for what sensor would be of any use here, they order a few sample sheets of these printed sensors and a small gadget that clips on the sheet and records the measurements of the sensors. After dipping them on a dozen coffees with different mixtures of sweeteners inside, they plug the gadget to their computer and get a spreadsheet of sensor values for each dipping session. They simply append the label they decided each sweetener level corresponds to and pass the spreadsheet to the system. They decide that the reported accuracy and area are manageable order the resulting circuit to be printed on a batch of flexible patches they can stick to the inside of the glasses.

4 Background information - Prerequisites

4.1 Uses of printed electronics

The usage of printed electronics most people may be familiar with in their everyday lives is the membrane used to detect key presses in most non-mechanical keyboards, or perhaps windshield defrosters.

Other usages include:

- Sensors: flexible, biodegradable and stretchable sensing elements enable the efficient monitoring of many processes. A variety of properties of the world can be measured by printed sensors, including temperature, touch, strain, gasses, humidity, light levels and presence of certain chemicals. The flexibility and non-toxicity is especially relevant for medical monitoring, so biosensors have received a lot of attention, with some (for example ,printed seizure detecting patches) already commercially available.
- RFID: RFID (Radio Frequency Identification) is a wireless technology reader, enabling seamless object identification and tracking through unique identification codes stored in the tags. The goal of printed RFID is to replace current methods for identifying goods with smart labels. RFID tags are usually passive and don't require a power supply. They can be cheaply made with any common printing method. They have been shown to operate on 5G and WLAN frequencies, and can even have sensor capabilities. Currently mostly used in ticket fares and anti-shoplifting.
- Energy harvesting: Printed batteries can only provide power to the functional parts of printed circuits for a limited time, and can take up a significant portion of the circuit's area. In order to enable greater autonomy to deployed printed electronics the ability to harvest energy from the environment is crucial. Printed harvesters can draw power from radio signals, vibrations and most commonly, light. Printed photoelectric/solar cells have also drawn a lot of interest outside the realm of harvesters for small circuits, since while their performance doesn't reach the levels of rigid silicon solar cells they can be deployed in a wider selection of spaces, including wearables.
- Lighting: LEDs have become the predominant light source, in place of the energy wasteful incadecent lamps and the environmental minefields of fluorescent lighting. OLEDs further increase the energy savings and produce softer and more uniform lighting. Printing seems like a promising solution for low cost manufacturing of OLEDs with competitive luminous efficiency

and enable them to cover large areas. Paper thin light panels have been demonstrated that way.

- Displays: Displays are one of the more mature aspects of printed electronics, with large 4K printed OLED displays are commercially available. They enable flexible displays, that have many applications in consumer electronics and wearables and thus are a 5 billion dollar market. Even if the flexible display is not fully printed, printed electronics can offer it additional features. QLED displays may also one day be printed if printing accuracy keeps increasing.
- Wearables: Wearable electronic devices are already very popular, such as smart watches or hearing aids, or NFC rings. Printed electronics have much to offer to the space thanks to their flexibility. Conductive materials have been developed that can be printed on fabric and withstand washing with detergent, allowing electronics to be embedded in regular pieces of clothing. Printed sensors can be used for activity tracking, one of the most popular features of today's smartwatches, or health monitoring, with printed patches for seizure detection already on the market. One can also imagine they would be of interest to the fashion industry.

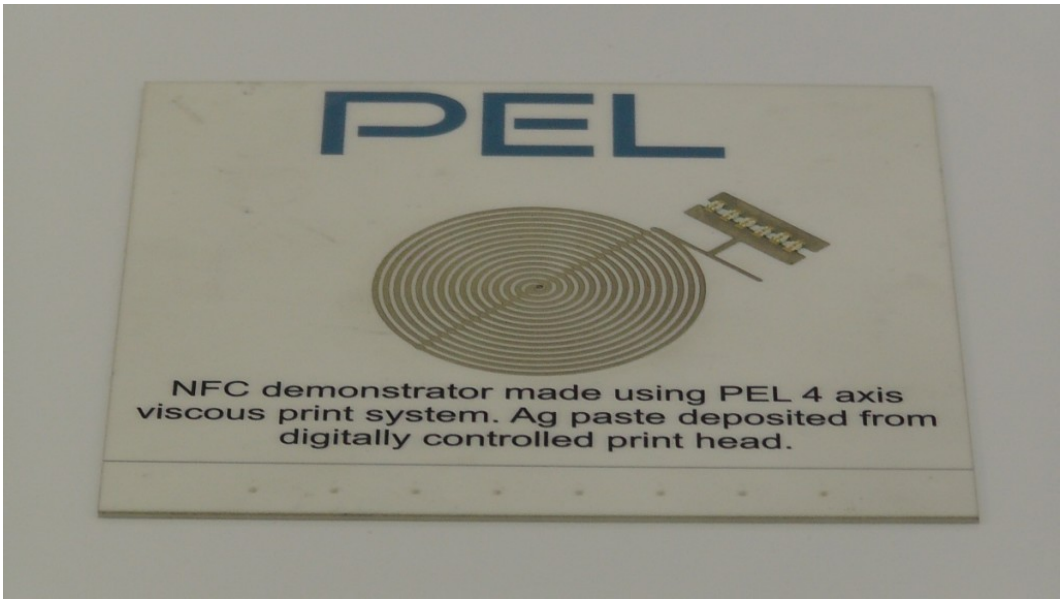

Figure 3: Printed NFC demontration. Source: PRINTED ELECTRONICS LTD

Figure 4: Printed circuit on the membrane of a common keyboard. Source: Paulo Maluf

4.2 Manufacturing methods

Printed electronics are manufactured using techniques from the graphic print industry. They are split into contact or R2R printing techniques that use a template and contactless that don't. Multiple printing steps are required for the multiple layers of the circuit. Contact printing techniques include:

- Gravure: In gravure, the printing cylinder gets engraved with the template and is partly submerged in ink during the process, with a blade discarding excess ink. This only leaves ink in the template parts, which is transfered to the substrate under pressure. Gravure can print in high resolution and speed compared to other methods, but the cost of engraving the cylinder makes it only useful for vary large batches.

- Offset: In offset printing the shape of the template is deposited on a cylinder with an ink accepting substance and the negative of the template is covered with ink repelling substances. That way only the shape of the template absorbs ink from an ink roller, and then gets transferred onto the substrate via an intermediate cylinder.
- Flexography: The template is embedded onto a flexible plate that is wrapped around a printing cylinder such that parts of the shape raised. Ink applied to this cylinder only gets transferred to a second cylinder, and then the substrate, if it is on the raised parts that correspond to the template. It can support both non-porous and porous substrates.

- Screen printing: A "screen" in this case is a close-knit fabric, such that ink can pass through only by applying pressure. A stensil of the template is placed on top of the screen and a blade pushes ink througth the uncovered parts onto the substrate. Screen printing is the simplest technique of the bunch and can create thicker layers and print on curved surfaces. It suffers from lower resolution compared to other methods.
- Pad printing: Ink gets onto an engraving of the template. A soft pad is then pressed on it and transfers the ink with the desired shape to the substrate. It can print on surfaces of 3D objects.

Contactless techniques include:

- Inkjet: Ink is dropped onto the substrate from tiny spouts. Either there are enough spouts to cover the width of the print or they can be moved around to do so. It does not require large equipment and different designs can be printed in high resolution without complications in changing templates,

making it ideal for printing on demand. It's main drawback is it's printing speed. Continuous stream inkjet has a stream of ink be directed onto the substrate or to a trash bin depending on design information. It is can print larger batches than Drop-on-Demand inkjet, but with five times lower resolution. DoD controls whether ink will flow using a valve, so ink is not wasted. It is deployed at smaller scales than Continuous stream.

- Aerosol: The ink is atomised into a fine mist via compressed air or ultrasound, accelerated and sprayed onto the substrate. It can be used on curved surfaces and can provide even smaller feature sizes than inkjet, but is prohibitively slow.

Additionally methods like vacuum deposition, in which evaporated ink coats a surface in a vacuum, or dip pen nanolithography, in which an atomic force microscope applies the ink very precisely on the substrate, are sometimes considered included in the printed electronics umbrella, and although they can achieve smaller feature sizes they require specialised equipment and are not as cost friendly as the traditional printing methods and thus less relevant.

Figure 5: Dimatix DMP-2850 Materials Printer. Source: FUJIFILM

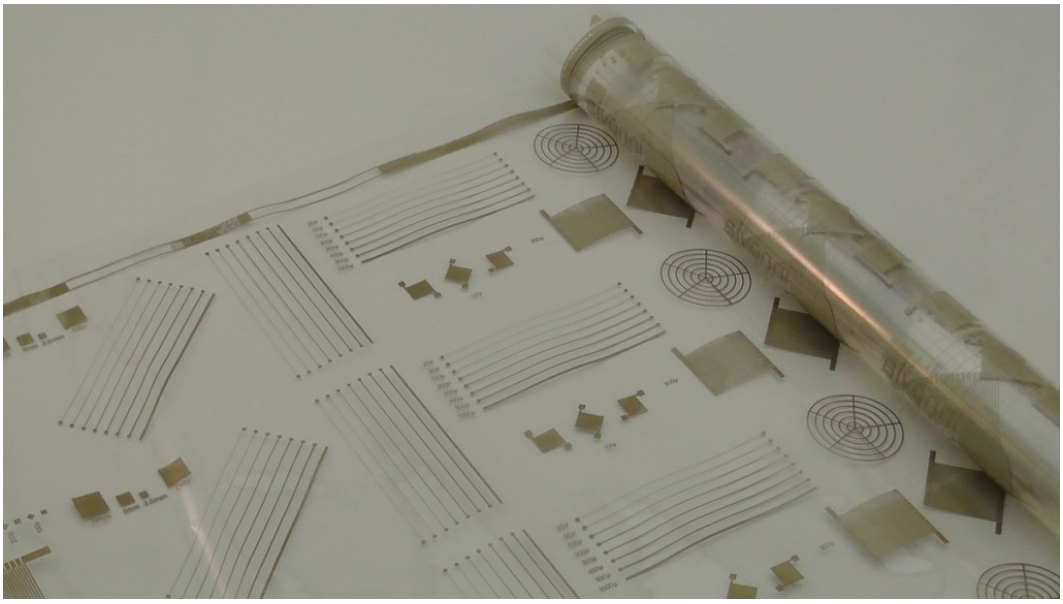

Figure 6: Long sheet of circuits printed on a Roll-to-Roll system. Source: PRINTED ELECTRONICS LTD

4.3 Inks

In order to implement functional circuits inks with conducting, semiconducting and dielectric properties are needed. They usually consist of nanoparticles of materials with these properties mixed with solvent to the desired viscosity and other additives to enable the printing process. Both organic and inorganic materials can be used.

- Conducting inks: The majority of materials for conducting inks are metal nanoparticles, the most common being silver. Although silver is in the category of precious metals silver ink is not awfully expensive, with pens of conductive silver ink going for less than 4 dollars. Other metals used are gold, aluminum and copper. Copper and aluminum inks suffer much worse ageing than silver ones. Organic inks are often based on carbon nanotubes or graphene. Cheaper polymers are also used, despite their inferior conductivity. The most popular is PEDOT:PSS. Ceramic materials are also used in conducting inks, mainly indium tin oxide(ITO), although it is an expensive material.
- Semiconducting inks: The most common inorganic materials used are silicon and germanium and of the organic ones most are again CNT or graphene based. Both p-type and n-type materials can be produced from those, although p-types have historically be much higher performing. (The opposite holds true for electrolyte gated transistors.)
- Dielectric inks: The dielectric layer needs to be thicker than the conducting and semiconducting layers in order for charge not to leak through it. Substrate materials, ceramic oxides and polymers can be used as the active ingredient.

4.4 TinyML

Edge computing enables applications where data processing is location sensitive. It provides greater security, privacy and availability guarantees to the end users. It is a fundamental component of the IoT market, that can reduce the dependance on cloud systems. The main bottleneck to it's addoption spread are the resource contstraints it imposes.

To deal with the demand of running machine learning applications on the edge for inteligent devices, traditional architectures are too bloated to make the cut. Many models nowadays demand computing capabilities out of reach for even the most high-end consumer hardware, let alone low power devices. TinyML is the field of optimising machine learning architectures to run on ultra resource constrained systems, typically no more than a few milliwatts.

Multidisciplinary work is demanded for this undertaking, as both the ML algorithms, the software and the hardware that supports them must accomodate these constraints while not compromising the accuracy of the models to a significant degree. Roughly the constraints at play are energy efficiency, processing capacity, memory space and production and engineering costs. It should be emphasised that the concern is with the inference step of ML although enabling the training phase on edge hardware is also it's own niche endeavor.

Since pretrained ML models cannot be run on these terms by default, end to end pipelines are required from data acquisition to inference, giving rise to the field of TinyML-as-a-Service or TinyMaaS. Special precausions must be taken at every intermediate step to lead to a runtime model lightweight enough.

Some approaches to the problem include:

- An alternative to common ML models that can function at lower energy budgets is Hyperdimensional computing (HDC). In this paradigm. It leverages the propery of high-dimentional spaces that randomly sampled vectors are almost certainly close to orthogonal to each other to represent classes and features as ensembles of hypervectors. Samples are mapped to querry hypervectors during inference, which are then compared to the class encoders and the most similar is declared as the predicted class. Vectors usually have binary elements of 1 or -1 to reduce requirements. It can handle noisy and/or incomplete data, which is a big plus. Unfortunately the count of dimensions needed for many problems are so large we cannot make savings by this method.

- One of the most ubiquitous methods in the field is constrained neural architecture search (NAS). Neural architecture search examines a search space of different architectures for different hyperparameters. An algorithm tries to locate the best possible architecture to maximise model performance on the objective function. An evaluator examines the trade-offs between accuracy and efficiency on deployement, given declared constraints of memory, energy etc. It may consider one or many target models on a single or multiple platforms. Both the search space and the search algorithm are hardware-aware. It is a multi-objective optimisation problem that is usually implemented as a multi-stage single-objective optimisation problem. Running the search is very time consuming but results outperform most manually designed networks.
- An obvious approach to handling the memory constraint issue is using data compression techniques on the ML model. A key approach that has demonstrated 15-40x compression factors are Kronecker Products (KP). Large accuracy penalties may occur however, and a method called doped Kronecker product (DKP) leverages co-matrix adaptation to try and remedy those.
- Once-for-all network is the name of a method for decoupling the architecture search from model training. OFA can find specialised subnetworks that largely maintain the accuracy of the full accuracy larger model without the need for training. A recent progressive shrinking algorithm (PSA) reduces depth and width of layers while retaining a higher final accuracy than equivalent general pruning.

- Over-parameterization is the property of a neural network where redundant neurons do not improve the accuracy of results. This redundancy can often be removed with little or no accuracy loss. Fully connected deep neural networks require N^2 connections between neurons. Network pruning removes parameters that don't impact accuracy by a large amount. A common case where this can easily be done is when parameters are either zero or sufficiently close to it. Similarly when the parameter values are redundantly duplicated. It can be applied at any granularity, from individual connections, neurons to entire layers. When a pruning procedure results in the neural network losing it's symmetrical structure it is referred to as unstructured pruning, otherwise it is structured pruning. Unstructured pruning results to sparse weight matrices that general processors do not execute efficiently. Retraining a network after pruning parameters that weren't contributing enough can allow it to reach higher accuracies than before. Even pruning a randomly initialised network without training it before or after can result to a decent accuracy. Pruning is split into static pruning and dynamic pruning. Static pruning removes neurons between the training and inference stages, while in dynamic pruning it happens during the runtime. Usually all of the model's weights are available in runtime and a controller conditionally includes or excludes parts of the network from the computation based on detecting certain feature patterns. This controller is often also a trained machine learning model. Criteria for which elements to prune include brute-force pruning, where the entire model is searched element by element to find ones that don't affect the outcome. Norms of weight vectors may be used to prune neurons, in particular the popular L1 norm in the LASSO method. Optimal Brain Damage uses the second derivative or Hessian matrix of the loss function to locate unneeded weights and was succeeded by the similar Optimal Brain Surgeon. Calculating these derivatives is too computationally expensive to be applied to larger networks. Average Percentage Of Zeros is a method to judge if the outputs of a neuron are usually contibuting to the result. Penalty based pruning introduces constraints during training to result to more weights being near zero so they can be pruned. Feature selection removes input features of the model that are not utilised sufficiently. It prevents overfitting and accelerates training. Another method clusters hidden features of a layer and removes those that are close enough to be reduntant. Iteratively pruning a network then retraining it with only the remaining elements allows to remove a much larger percentage of parameters without damaging performance. Knowledge distillation with the original network as the teacher and the pruned network as student can be used to recover lost accuracy.

- Knowledge distillation is a process of training a smaller, shallower student network to much the output logits of a larger, more capable teacher network that has been trained to satisfactory accuracy. More advanced variants include ensembles of small networks each trying to match the results of the congregated ensemble, or self-distillation in which shallower layers of a deep neural network try to learn to match the more complicated features of the deeper layers. If the size difference between the student and teacher networks is too large an intermediate size teacher assistant network gets the teacher's answers distilled into it and subsequently distills them to the student.
- Quantization is the process of reducing the numerical precision of values in the model. Networks are typically using 32 bit floating point numbers during training[26]. The most common quantization targets for those are either 8-bit or 4-bit integers. In many cases the network does not utilise this level of precision to it's full extend. Reducing the precision in these cases can relieve the computational burden associated with negligible accuracy sacrifices. Quantization-aware training is a process in which the full precision network is fine-tuned of retrained into the reduced form. It succeeded quantizing the network only after the training process is completed. When the method is pushed to the limit, precision is reduced down to 1 bit. Networks with a single bit of precision are called Binary Neural Networks(BNNs).

4.5 Binary Neural Networks

BNN is the term for neural networks that have both activations and weights in 1 bit precision in all hidden layers. Input layers ought to have higher precision inputs so the network can receive sufficient information for classification to be possible, and output layers of classifiers have their activations compared to each other to decide on the predicted class, so they cannot be binarized. The most common domain for BNNs are Convolutional Neural Networks(CNNs). They were independently presented in 2016 by [11] and [21].

Beyond reducing the storage size required for the weights $32\times$ compared to a full precision 32-bit network of the same architecture, computation costs are significantly dropped too since the multiply-accumulate(MAC) operations can be carried out by XNOR and popcount operations. This can lead to up to a $58\times$ improvement in speed.

During training, higher precision underlying weights are used to make learning more robust. In the forward propagation phase these more precise weights, W, and the activations from the previous layer I are binarized using the sign function:

$$sign(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0 \end{cases}$$

The binary operations $(-1, 1, *)$ and $(0, 1, \odot)$ are isomorphic, so the multiplication of weights with activations is done using the XNOR operation when the binary values {-1,1} are encoded into the logic values {0,1} to be stored in a bit.

This mapping can be represented by the linear transformation $f(x) = \frac{x+1}{2}$, since $f(-1) = 0$ and $f(1) = 1$. The accumulation and subsequent binarization of the activation-weight products $sign(I) * sign(W)$ can be calculated by performing a popcount operation, which returns the count of bits in a given collection that are 1, and comparing the result with a threshold to binarize it.

During backward propagation, an approximation of the activation function $sign$ needs to be used since $sign$ has a derivative of 0. The most common method is known as straight-through estimation(STE). In STE the sign function is approximated by:

$$STEsign(x) = \begin{cases} +1, & \text{if } x \geq 1 \\ x, & \text{if } 1 \geq x \geq -1 \\ -1, & \text{if } x \leq -1 \end{cases}$$

which has a derivative of:

$$\frac{dSTEsign}{dx} = \begin{cases} 1, & \text{if } 1 \geq x \geq -1 \\ 0, & elsewhere \end{cases}$$

Updates are made to the underlying higher precision weights, and their binarizations are used for the forward pass.

4.6 Datasets

The datasets chosen to train models for and implement are the ones used by [24]. That way results for model accuracy and area / power requirements can be compared with other approaches in the literature. Like in those papers, categorical features were removed from the datasets, leaving only inputs from sensors, since they are all the actual printed system would have access to (this assumption may be circumvented, but this is beyond the current scope). Note that the feature selection may not be the same as the prior papers, since the pieces of data they kept were not documented. All of them were taken from the UCI machine learning repository[14].

A short description of the datasets:

- Arrythmia[16]: Diagnosis of cardiac arrhythmia from 12 lead ECG recordings.
- Cardiotocography[6]: Diagnosing problems in the heartrate of unborn infants.
- Pendigits[1]: Classification of written digit from a series of 8 pressure signals from touch sensors.
- Human activity recognition(HAR)[2]: Classification of the type of movement a person is making(standing, climbing stairs etc) using accelerometers from cellphones on their waists.
- Gas Identification[15]: Classification of gas presence using chemical sensors.
- Wine Quality(White wines)[10]: Estimating the percieved enjoyment of various white wines based on acidity and mineral traces.
- Wine Quality(Red wines)[10]: Equivelant to the above for red wines.

The datasets use inputs from sensors that at least approximately correspond to ones that have been demonstrated possible to manufacture by printing. The complete system including both sensors, classifier and power supply could thus somewhat realistically be physically implemented, and not be very far from an actual usecase of the technology.

Sensor	Dataset
Electrocardiography sensor on paper[8]	Arrythmia
Electrocardiography sensor on paper[8]	Cardio
Printed movement sensor	Human activity recognition
Printed gas sensor[12]	Gas identification
Printed piezoelectric sensor[27]	Pendigits
Printed pH sensor[20], Inkjet mineral sensor[19]	Wine Quality(White)
Printed pH sensor[20], Inkjet mineral sensor[19]	Wine Quality(Red)

[1] F. Alimoglu, E. Alpaydin, Methods of combining multiple classifiers based on different representations for pen-based handwritten digit recognition, in: Proceedings of the Fifth Turkish Artificial Intelligence and Artificial Neural Networks Symposium (Tainn 96), Citeseer, 1996.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. Reyes-Ortiz, A public domain dataset for human activity recognition using smartphones, in: Esann, 2013.

[3] G. Armeniakos, G. Zervakis, D. Soudris, M.B. Tahoori, J. Henkel, Model-to-circuit cross-approximation for printed machine learning classifiers, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. (2023).

[4] G. Armeniakos, G. Zervakis, D. Soudris, M.B. Tahoori, J. Henkel, Co-design of approximate multilayer perceptron for ultra-resource constrained printed circuits, IEEE Transactions on Computers. (2023).

[5] G. Armeniakos, G. Zervakis, D. Soudris, M. Tahoori, J. Henkel, Cross-layer approximation for printed machine learning circuits, in: Design, Automation Test in Europe Conference Exhibition (Date), 2022.

[6] D. Ayres-de Campos, J. Bernardes, A. Garrido, J. Marques-de Sa, L. Pereira-Leite, Sisporto 2.0: A program for automated analysis of cardiotocograms, Journal of Maternal-Fetal Medicine. 9 (2000) 311–318.

[7] K. Balaskas, G. Zervakis, K. Siozios, M.B. Tahoori, J. Henkel, Approximate decision trees for machine learning classification on tiny printed circuits, in: 2022 23rd International Symposium on Quality Electronic Design (Isqed), IEEE, 2022: pp. 1–6.

[8] E. Bihar, T. Roberts, M. Saadaoui, T. Hervé, J.B. De Graaf, G.G. Malliaras, Inkjet-printed pedot:PSS electrodes on paper for electrocardiography, Advanced Healthcare Materials. 6 (2017).

[9] N. Bleier, M. Mubarik, F. Rasheed, J. Aghassi-Hagmann, M. Tahoori, R. Kumar, Printed microprocessors, in: 2020 Acm/Ieee 47th Annual International Symposium on Computer Architecture (Isca), IEEE, 2020: pp. 213–226.

[10] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis, Modeling wine preferences by data mining from physicochemical properties, Decision Support Systems. 47 (2009) 547–553.

[11] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, Y. Bengio, Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1, arXiv Preprint arXiv:1602.02830. (2016).

[12] J. Dai, O. Ogbeide, N. Macadam, Q. Sun, W. Yu, Y. Li, W. Huang, Printed gas sensors, Chemical Society Reviews. 49 (2020) 1756–1789.

[13] M. Douthwaite, F. García-Redondo, P. Georgiou, S. Das, A time-domain current-mode mac engine for analogue neural networks in flexible electronics, in: 2019 Ieee Biomedical Circuits and Systems Conference (Biocas), IEEE, 2019: pp. 1–4.

[14] D. Dua, C. Graff, UCI machine learning repository, (2017).

[15] S. Feng, F. Farha, Q. Li, Y. Wan, Y. Xu, T. Zhang, H. Ning, Review on smart gas sensing technology, Sensors. 19 (2019) 3760.

[16] H.A. Guvenir, B. Acar, G. Demiroz, A. Cekin, A supervised machine learning algorithm for arrhythmia analysis, in: Computers in Cardiology 1997, IEEE, 1997: pp. 433–436.

[17] IDTechEx, Flexible & printed electronics 2023-2033: Forecasts, technologies, markets, 2023.

[18] K. Iordanou, T. Atkinson, E. Ozer, J. Kufel, J. Biggs, G. Brown, M. Lujan, Tiny classifier circuits: Evolving accelerators for tabular data, (2023).

[19] M. Jelbuldina, H. Younes, I. Saadat, L. Tizani, S. Sofela, A. Al Ghaferi, Fabrication and design of cnts inkjet-printed based micro fet sensor for sodium chloride scale detection in oil field, Sensors and Actuators A: Physical. 263 (2017) 349–356.

[20] M. Jose, S.K. Mylavarapu, S.K. Bikkarolla, J. Machiels, S. KJ, J. McLaughlin, W. Deferme, Printed pH sensors for textile-based wearables: A conceptual and experimental study on materials, deposition technology, and sensing principles, Advanced Engineering Materials. 24 (2022).

[21] M. Kim, P. Smaragdis, Bitwise neural networks, arXiv Preprint arXiv:1601.06071. (2016).

[22] A. Kokkinis, G. Zervakis, K. Siozios, M.B. Tahoori, J. Henkel, Hardware-aware automated neural minimization for printed multilayer perceptrons, in: 2023 Design, Automation & Test in Europe Conference & Exhibition (Date), IEEE, 2023: pp. 1–2.

[23] H. Ling, D. Koutsouras, S. Kazemzadeh, Y. van de Burgt, F. Yan, P. Gkoupidenis, Electrolyte-gated transistors for synaptic electronics, neuromorphic computing, and adaptable biointerfacing, Applied Physics Reviews. 7 (2020) 011307.

[24] M. Mubarik, D. Weller, N. Bleier, M. Tomei, J. Aghassi-Hagmann, M. Tahoori,

R. Kumar, Printed machine learning classifiers, in: Annu. Int. Symp. Microarchitecture (Micro), 2020: pp. 73–87.

[25] E. Ozer, J. Kufel, J. Biggs, G. Brown, J. Myers, A. Rana, C. Ramsdale, Bespoke machine learning processor development framework on flexible substrates, in: 2019 Ieee International Conference on Flexible and Printable Sensors and Systems (Fleps), IEEE, 2019: pp. 1–3.

[26] V. Sze, Y. Chen, T. Yang, J. Emer, Efficient processing of deep neural networks: A tutorial and survey, Proceedings of the IEEE. 105 (2017) 2295–2329.

[27] S. Tuukkanen, S. Rajala, A survey of printable piezoelectric sensors, in: 2015 Ieee Sensors, IEEE, 2015: pp. 1–4.

[28] D. Weller, N. Bleier, M. Hefenbrock, J. Aghassi-Hagmann, M. Beigl, R. Kumar, M. Tahoori, Printed stochastic computing neural networks, in: Design, Automation Test in Europe Conference Exhibition (Date), 2021: pp. 914–919.

[29] D. Weller, M. Hefenbrock, M. Tahoori, J. Aghassi-Hagmann, M. Beigl, Programmable neuromorphic circuit based on printed electrolyte-gated transistors, in: 2020 25th Asia and South Pacific Design Automation Conference (Asp-Dac), 2020: pp. 446–451.