Detecting technological maturity from bibliometric patterns

Katherine Cauthen[a,*], Prashant Rai[b], Nicholas Hale[c], Laura Freeman[c], and Jaideep Ray[b]

[a] Sandia National Laboratories, 1515 Eubank Blvd. SE, Albuquerque, NM 87123, USA
[b] Sandia National Laboratories, 7011 East Avenue, Livermore CA 94550-0969, USA
[c] Virginia Tech Applied Research Corporation, 900 N. Glebe Road, Arlington, VA 22203, USA

Declarations of interest: None

* Corresponding author at: Sandia National Laboratories, 1515 Eubank Blvd. SE, Mail Stop 1137, Albuquerque, NM 87123. (505) 845-7002.

Email addresses: kcauthe@sandia.gov (K. Cauthen), nshale2@gmail.com (N. Hale), laura.freeman@vt.edu (L. Freeman), prashantrai.241@gmail.com (P. Rai), jairay@sandia.gov (J. Ray).

Abstract

The capability to identify emergent technologies based upon easily accessed open-source indicators, such as publications, is important for decision-makers in industry and government. The scientific contribution of this work is the proposition of a machine learning approach to the detection of the maturity of emerging technologies based on publication counts. Time-series of publication counts have universal features that distinguish emerging and growing technologies. We train an artificial neural network classifier, a supervised machine learning algorithm, upon these features to predict the maturity (emergent vs. growth) of an arbitrary technology. With a training set comprised of 22 technologies we obtain a classification accuracy ranging from 58.3% to 100% with an average accuracy of 84.6% for six test technologies. To enhance classifier performance, we augmented the training corpus with synthetic time-series technology life cycle curves, formed by calculating weighted averages of curves in the original training set. Training the classifier on the synthetic data set resulted in improved accuracy, ranging from 83.3% to 100% with an average accuracy of 90.4% for the test technologies. The performance of our classifier exceeds that of competing machine learning approaches in the literature, which report an average classification accuracy of only 85.7% at maximum. Moreover, in contrast to current methods our approach does not require subject matter expertise to generate training labels, and it can be automated and scaled.

Keywords

Technology life cycle, machine learning, artificial neural network, data augmentation

1 Introduction

Research and Development (R&D) managers must make strategic decisions about when to invest in, hold or divest from a technology. The maturity of a technology plays an important role in this decision as it governs the availability, cost, and obsolescence of the said technology. R&D managers obtain this information from standardized short intelligence reports which cover a myriad of topics, technology maturity being one. Many open-source proxies of technological maturity exist, and scanning them to produce a quantitative measure, in an economical and timely fashion, requires that the process be automated and implemented via information technology (Albert, 2015). This paper presents a method to do so, using statistical/machine learning.

Commented [GS1]: Not exactly sure what this means, what is information technology? Perhaps confusing because previous sentences are discussing technology information.

Technology Life Cycles (TLC) are often characterized using an S-curve (a sigmoid). The abscissa ("x-axis") is usually time but could be any measure of effort required to develop the technology e.g. funding. The ordinate ("y-axis") is a measure of maturity (e.g., the number of products using the technology or some performance parameter). The S-curve illustrates that technologies emerge slowly (*emergent* phase) but accelerate into a *growth* phase. At a point, the technology matures, and slows down (the *saturation* phase), and then finally plateaus (the *decline* phase). A similar process is seen in Product Life Cycles (PLC) curves, though they are usually cast in terms of incremental changes per unit time i.e., they are a time-derivative of the S-curve, and are called the “bell-curve.” We will use the bell-curve formalism in this paper. In this formalism, the peak of the curve separates the growth from the saturation phase.

Commented [GS2]: Good description, I could follow what was being described, but a graphic of a sigmoid w/ axis labels and various phases highlighted could be useful.

There are many methods to study the S- or bell-curves; see (Taylor & Taylor, 2015) for a review. Albert’s Ph.D. thesis (Albert, 2015) presents a concrete realization of Sommerlatte and Deschamps’s TLC evolution model (Sommerlatte & Deschampes, 1986) as a transition between four phases (emergence-growth-saturation-decline). Each of the phases can be characterized using 9 traits (Table 2.12, in (Albert, 2015)) but a time-series of scientific publications and patents, collated annually, are two traits that change with the four phases i.e., the two time-series follow a bell-curve, and have the same four phases. In this paper we will use an annual time-series of scientific publications as a proxy for the TLC. Then, using 22 such TLCs as a training corpus, we will construct a model that given a test TLC, will classify the years as belonging to the emergent or growth phase. Our model stipulates that the TLCs not be labeled by subject matter experts (SMEs) into their four phases, but rather be labeled by a model that keeps a certain fraction of the published papers in the emergent phase.

As should be clear, the use-case for this classifier is in the widescale, automated scanning of a vast number of new technologies, with the aim of detecting transition from emergence to growth. This transition should coincide, approximately, with the start of commercial interest in the technology. We will not address transitions to saturation or decline, as by then, the technology is widely known

Our approach is based on the hypothesis that the transition between emergence and growth is encoded in the shape of the TLC curve itself. Specifically, we hypothesize, that the derivatives of the TLC curve, suitably normalized, assume very different values for the emergent and growth phases, and these values are universal across technologies i.e., the transition boundary could be learned from our training corpus. We will “learn” this boundary in the form of a binary classifier. The approach, if successful, could simplify the detection of emergent and growing technologies, conditioned on open-source information. This is the first contribution of this paper.

Commented [GS3]: Could be interesting to state how comfortable an assumption this was given real technology maturation data

The compilation of a training corpus for training the classifier is quite difficult. Since the TLC curves constituting the training corpus should contain all four phases, it will, perforce, contain obsolete technologies. This eliminates recent and many relevant technologies. In addition, technology maturity cannot be measured directly and therefore requires some proxy measure, such as publication count, which precipitates other challenges. Government-funded scientific and engineering research, which necessitated regular publications as a measure of funding performance, only became common in the 1950s. In addition, many TLC curves, which are based on technology maturity proxies, show spurious artifacts. For example, the time-series of publications referring to a smallpox vaccine shows two peaks, one during the 1970s when the smallpox eradication program was being pursued and another one that spanned 1998-2010, when there was interest in bioterrorism. However, these later papers dealt mostly with mathematical models to investigate the most efficient way to vaccinate a population. The actual technology for manufacturing or improving the vaccine was rarely considered and it is debatable if these papers should be considered as a representation of the TLC. It is not often that one can discern and remove such artifacts manually, and consequently constructing the training corpus can be difficult, and somewhat subjective, resulting in a small one. A classifier trained on a small corpus suffers from two shortcomings – (1) it assimilates little information, leading to a classifier that predicts inaccurately and (2) the training process is difficult, leading to a classifier that is sometimes unstable. There is nothing much that can be done about the first shortcoming without expanding the training data, but the second shortcoming, being entirely numerical in nature, might admit an algorithmic improvement.

Commented [GS4]: Interesting paragraph! I like that data complexities are being address directly up front.

The second contribution of this paper is a method to make a large synthetic training corpus using our original scarce one. The hypothesis is that the TLC curves are quite simple and could be encoded in a low-dimension space. If true, then our scarce training corpus could adequately define a distribution in low-dimensional space, implying that randomly weighted linear combinations of the TLC curves could yield a large synthetic dataset. This dataset would not contain any new information (being derived from the original one), but it could render the training process easier. A classifier trained on the synthetic dataset should prove to be as accurate, and perhaps slightly better than the original one. We will test this hypothesis using held-out TLC curves i.e., TLC curves that were not used to construct the large synthetic training corpus. With the inclusion of synthetic training data, we found that the classification accuracy of the held-out TLC curves improved, and moreover exceeded the performance that has been reported in the literature for other machine learning approaches to technology maturity classification.

Commented [GS5]: I'm assuming the held out TLC curves were not used in anyway to generate the synthetic training data?

The paper is structured as follows. In Section 2, we will review literature on various ways of gauging and modeling technological maturity. In Section 3, we will formulate the statistical problem. We will extract, smooth and label the TLC curves. In Section 4, we postulate the classifier as an artificial neural network (henceforth, *neural net*), train and test it and verify our first hypothesis. In Section 5, we will construct the synthetic dataset, retrain a new classifier (also a neural net) and check its performance on held-out TLCs to verify our second hypothesis. The paper concludes with a summary of the two contributions and proposals to extend the research.

2 Literature Review

2.1 Technology Life Cycle Models

Historically researchers have used expert knowledge to assess technology maturity. One common approach is the Delphi method in which experts iterate through several rounds of elicitation with feedback until their opinions converge (Dalkey & Helmer, 1963). The results of the Delphi method and other consensus methods can be skewed by experts' imperfect, qualitative knowledge of the technology domain. Additionally, relying on subject matter expertise is costly and time-consuming (Lemos & Porto, 1998). In response more quantitatively rigorous, automated approaches have been developed.

Several models for the life cycle of a technology have been developed (Sommerlatte & Deschamps, 1986; Ford & Ryan, 1981; Ansoff, 1984; Linden & Fenn, 2003), but the fields of economics and management have broadly adopted two general curve models (Nieto et al., 1998). The first is an S-Shaped curve of cumulative innovation over time or R&D expenditures (Merino, 1990; Ernst, 1997; Wu et al., 2011; Gao et al., 2013). Cumulative innovation represents the total sum of innovations up to that point in time. An S-shaped curve has been supported by the findings of several empirical studies (Achilladelis et al., 1990; Achilladelis et al., 1993; Achilladelis & Antonakis, 2001; Andersen, 1999). The second model is a bell-shaped curve of rate of innovation over time or R&D expenditures (Urban & Hauser, 1993; Nieto et al., 1998). Rate of innovation represents the number of innovations per unit time. Taylor and Taylor (2012) review the literature and argue that a bell-shaped curve can be applied to TLCs.

Both the S- and bell-shaped curves can be split into four distinguishable stages signifying introduction, growth, maturity, and decline of the technology (Taylor & Taylor, 2012). In the introduction stage there is a breakthrough and a technology's innovations slowly increase. As the technology grows the innovations increase more rapidly. Upon reaching the maturity stage innovation rate stagnates. Finally, in the decline stage innovations decrease as the technology is replaced by other emerging technologies (Kim, 2003).

Commented [GS6]: Innovation CDF vs innovation PDF?

2.2 Technology Life Cycle Indicators

A quantitative approach to modeling technology maturation requires some measure of technology maturity. Since technology maturity itself cannot be directly observed and measured, a variety of indicators have been used to serve as proxies for technology performance in modeling the TLC. Some researchers employ univariate indicators such as counts of patents or publications (Ramadhan et al., 2018; Lezama-Nicolás et al., 2018; Byun et al., 2018; Sick et al., 2018; Albert, 2015), news articles (Lezama-Nicolás et al., 2018), web search queries (Albert, 2015), start-up companies or product launches (Sick et al., 2018). Others leverage a suite of patent- or publication-related indicators (Gao et al., 2013; Kim et al. 2012; Lee et al., 2016; Su, 2018; Albert, 2015). Albert et al. (2015) perform text analytics on a corpus of blog text. Momeni and Rost (2016) extract patent citation paths, while others consider patent citations as a network instead (van der Pol & Rameshkoumar, 2018; Smojver et al., 2019). Other indicators that have been framed as networks include the collaboration network of co-authors on publications and patents (van der Pol & Rameshkoumar, 2018) and patent keywords (Smojver et al., 2019).

2.3 Quantitative Approaches to Assessing Technology Maturity

Various methodologies have been employed to model TLCs using these indicators. A classic approach is curve fitting, in which an S- or bell-shaped curve is fitted with a statistical distribution such as a Gompertz, logistic, Weibull, or normal (Stapleton, 1976; Franses, 1994; Ryu & Byeon, 2011; Sharif & Islam, 1980; Nagula, 2016). While curve fitting approaches allow for forecasting based upon a single indicator, subject matter expertise is still required for interpretation of the results, which allows room for unquantified bias and uncertainty and requires additional resources. Some researchers also argue that the accuracy and reliability of curve fitting methods is questionable (Haupt et al., 2007; Watts & Porter, 1997; Gao et al., 2013).

In order to avoid the inherent pitfalls of curve fitting, more recent advances use machine learning. Lee et al. (2016) present a single case study in which they use continuous hidden Markov models, an unsupervised machine learning method, to calculate a transition probability matrix between stages of the life cycle. However, most machine learning efforts in classifying the TLC stages have been supervised approaches. Kim et al. (2012) used decision trees to classify stages of technology evolution with 85.7% accuracy. Gao et al. (2013) demonstrated the efficacy of a k-nearest-neighbors classifier for a single technology. Linear Discriminant Analysis (LDA) has been successfully employed with up to 83.3% classification accuracy (Albert, 2015). A neural net approach to classifying TLC phases has also been developed (Ramadhan et al., 2018). It was trained and tested on only one technology, so its accuracy in classifying other technologies that may not have yet undergone the entire TLC is unknown. Machine learning approaches employed thus far have shown promise in learning the often non-linear patterns that drive the technology life cycle. However, these studies have either been demonstrated on only one technology and require subject matter expertise to label training data, been validated using only subject matter expertise, if at all, or in the case of Kim et al. (2012) and Albert (2015) proven to be less than 86% accurate over many technologies when classifying maturation stages. While these approaches are promising, work remains to be done.

Commented [GS7]: Specific reason to not use TLC here?

Commented [GS8]: Kind of specific and oddly focused, maybe better to word as "…have only achieved at most an 86% accuracy…" or something like that.

Other researchers have employed a variety of methods for determining the TLC stage. Albert et al. (2015) use fuzzy logic to map the outputs of sentiment analysis into TLC stages. Smojver et al. (2019) classify TLC stages by the degree distribution or growth of the graphs of patent citations and keywords. Other researchers do not explicitly classify TLC stages, but rather they characterize stages based on some result. For example, Momeni and Rost (2016) apply a forward-citation node pair algorithm to patent citations and characterize technology maturity stages based upon the resulting patent-development paths. Similarly, van der Pol and Rameshkoumar (2018) describe how patterns in graph topology of International Patent Classification (IPC) codes and collaboration networks vary by technology maturity stage. Su (2017) used ANOVA (analysis of variance) to determine which patent-related indicators are useful in distinguishing technology maturity stages.

Given the initial success of machine learning approaches we employed a neural net classifier. We sought to train this classifier on just one indicator for multiple training technologies so that it could be used to classify the TLC stage of any new technology.

Commented [GS9]: Why a neural net and not another ML based classifier? Maybe worth adding a sentence here, I understood why ML from previous text, but not why a NN.

Commented [GS10]: Again, maybe worth expanding this, and stating what the indicator is.

3 Preparing the Data

3.1 Data Extraction

A critical step in developing the machine learning classifier is the selection of the training and testing data. The TLCs, as approximated by publication counts, were extracted from the Elsevier Scopus database of scientific publications. The Scopus database was chosen to demonstrate the approach we propose in this paper because it is multidisciplinary, has high quality control standards for data, and has an easily accessible API. However, the approach need not strictly be applied to data extracted from the Scopus database, as the method is generalizable to many databases of scientific publications. The Title, Abstract, and Keywords (Author Assigned and Scopus Curated) fields were queried for the timeframe 1861-2019. We adopted a wide range in dates (all dates available in Scopus up to 2019) to ensure that full technology curves were captured in the data. The annual publication counts for each query were recorded and scaled by dividing by the total number of publications recorded in the entire Scopus database for that year. This scaling process is required to mitigate bias in the publication count indicator due to the ever-increasing number of total publications on the Scopus database, as shown in Figure 1.

Commented [GS11]: Coming back here to add comment, was there any consideration of how TLC timeframes have changed over time, like if the curve for a technology from 1940 is different than the curve of a technology from 1990? I'm assuming so, but just adding a note as it was something I was thinking about while reading this section.

*Subsequent graphics have time information for analyzed technologies

Commented [GS12]: Nice description and supporting figure

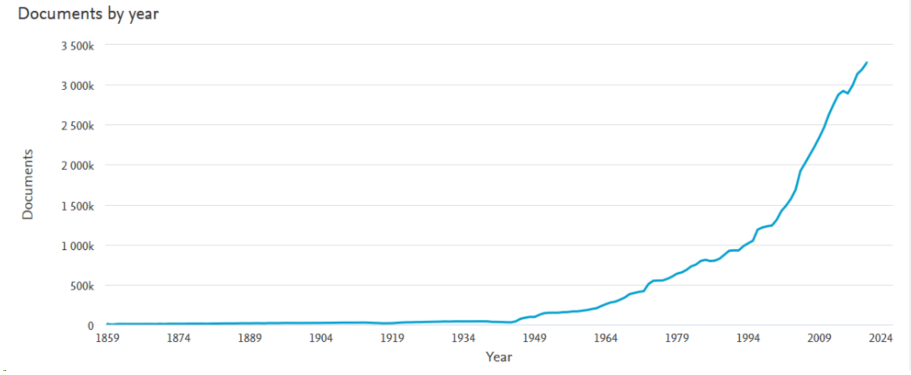


Figure 1. Total Scopus Publications

The technologies and their corresponding queries were selected and constructed by a group of Subject Matter Experts (SMEs) at Virginia Tech Applied Research Corporation and Sandia National Laboratories with backgrounds in Biophysics, Computer Science, Healthcare, Electrical, and Mechanical engineering. The technology subjects were anticipated by the SMEs to cover complete TLCs as to provide data containing emergence, growth, saturation, and decline. This selection method aimed at producing data representative of all four maturity phases, since many technologies produced partial cycles. The keywords listed in Table 1 were used as the queries. For technologies represented by multiple words (e.g., carbon nanotube) the keyword was enclosed in double quotes to form the query (e.g. "carbon nanotube").

Commented [GS13]: Previously SME informed methods were criticized, however, this is a different use case of SME information (data production vs data interpretation I believe), may be worth distinguishing this more explicitly.

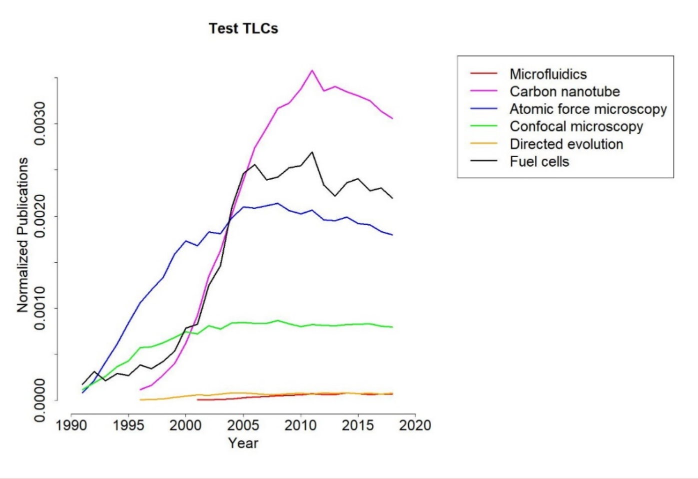
The full list of training and testing TLCs are given in Table 1. The testing TLC data were selected at random from the overall list of technologies. Per convention we opted to maintain a roughly 80%/20% split for training and testing. Figure 2 shows the normalized publication counts for all training and testing technologies.

Table 1. Training and Testing Technology Keywords

Training TLC data	**Testing TLC data**
airbag	microfluidics
high temperature superconductor	carbon nanotube

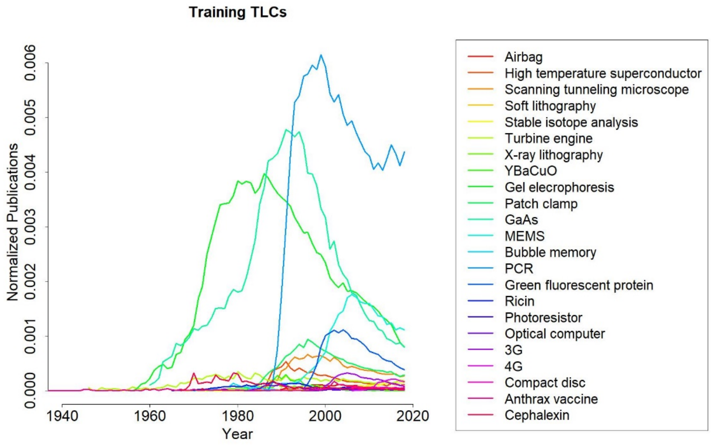
Commented [GS14]: Could reorganize table to have a training table and testing table, lots of empty space in the testing column

scanning tunneling microscopy	confocal microscopy
soft lithography	atomic force microscopy
turbine engine	fuel cell
x-ray lithography	directed evolution
YBaCuO	
gel electrophoresis	
patch clamp	
micro-electromechanical system	
optical computer	
bubble memory	
polymerase chain reaction	
green fluorescent protein	
ricin	
photoresistor	
3g	
4g	
cephalexin	
compact disc	
anthrax	
GaAs	
stable isotope analysis	



(a)

Commented [GS15]: Very cool graphics!



(b)

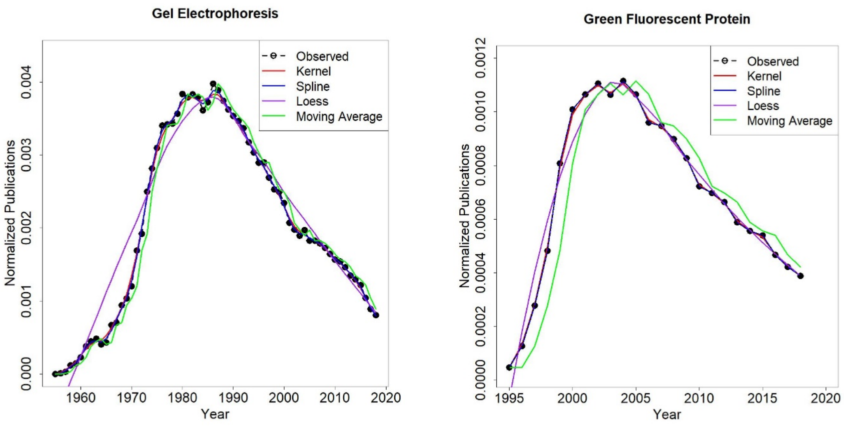
Commented [GS16]: Possibly just noise, but it does seem like some of the earlier technologies have higher normalized peaks relative to later technologies

Figure 2. Normalized publication counts for testing (a) and training (b) technologies

3.2 Data Smoothing

The extracted and normalized publication counts of the sample technologies resulted in 29 individual time series. Training and testing of the classifier required these time series to be smoothed so that derivatives could be computed. Three smoothing techniques were applied to the sample technologies and compared, including LOESS (LOcally Estimated Scatterplot Smoothing), moving average, spline, and kernel smoothing. The smoothing techniques were compared using RMSE (Root Mean Square Error) and a visual inspection to determine which technique resulted in smoothed values that were close to the original values without overfitting. The LOESS smoother proved to be effective over the greatest number of sample technologies, so it was applied to normalized publication counts to produce all the technology maturity curves for the classifier. Figure 3 shows two example TLCs and each candidate smooth curve.

Commented [GS17]: Depending on audience may be worth describing overfitting in this context



(a) (b)

Figure 3. TLCs smoothed using candidate techniques. (a) Gel electrophoresis technology curve is overfit with all smoothing techniques except for Loess smoothing. (b) Green fluorescent protein technology curve is overfit with spline and kernel smoothing, and LOESS smoothing results in a lower RMSE (5.71 x 10^{-5}) than that for moving average smoothing (1.07 x 10^{-4}).

Commented [GS18]: Nice description of graphics

3.3 Data Labeling

One goal of this research is to propose a machine learning based automated approach for classifying a technology into one of the phases of the technology life cycle (henceforth, the *maturity curve*). Since we are proposing a supervised approach to predict the maturity of a technology, we will require labeled training data. The methodology we used to prepare the training data consists of two sub-tasks. The first is to identify technologies which have gone through, ideally, all the four phases of the maturity cycle, or at least the phases in which we are most interested. The second task is to label the points in the smoothed time series by one of the four phases (emergence-growth-saturation-decline).

Commented [GS19]: First three phases right?

Unlike classical machine learning applications where labeling of data points may be trivial (e.g., labeling images of classification categories), labeling in this problem requires judgement of experts on specific technologies that have been identified. There are two potential problems with this labeling strategy. First, it may not be feasible, or even possible to identify experts with knowledge or expertise in labeling the data points as required. Second, such a labeling procedure will invariably lead to a subjective assessment i.e., two experts on a given technology may or may not agree with particular labels identified.

Commented [GS20]: Do SMEs see the technology maturation curves when labeling the data or do they use other sources?

In order to overcome these limitations, we first identify characteristics of phases in maturity cycle which must be satisfied by both the labeling procedure and the machine learning classifier. Let us consider the 'ideal' technology maturity curve $f(x)$ shown in Figure 4, which follows the probability distribution function of a normal random variable.

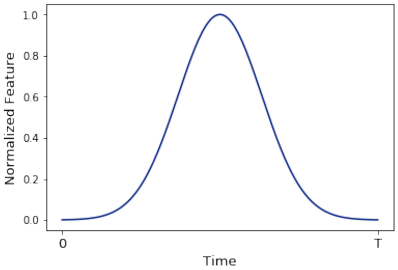


Figure 4: An ideal technology maturity curve. It follows probability density function of a standard normal distribution.

This curve corresponds to a hypothetical technology; however, it has the following characteristics useful for labeling of data:

- It depicts all the four phases of the maturity cycle
- It has a peak, called the stagnation point. Data points to the left of stagnation point belong to emergence and growth phases. Points to the right of stagnation belong to saturation and decline phases.
- The four phases of emergence, growth, saturation and decline follow sequentially in that order (i.e., there is no intermingling of points belonging to these phases)

Commented [GS21]: I'm curious about examples where this assumption is not satisfied for a technology and what drives it, not really feedback here, just curious

Our first objective is to design a strategy that labels the training data on this 'ideal' curve satisfying above characteristics. The maturity curve for a real technology will not be the same as Figure 3 but must necessarily have the same characteristics. Our second objective is to devise a strategy to map labeling from the ideal maturity curve to the maturity curve of any real technology. Next, we detail the procedure to achieve these two objectives.

Commented [GS22]: Maybe worth describing why this is necessary

In order to label points on the ideal maturity cycle, we use shape features of the curve that follow the desired characteristics mentioned above. Let t denote the time variable and $f(t)$ denote the feature, smoothened and normalized publication count in this case, that tracks the maturity of a technology. Let$\left(t_1, f(t_1)\right), \dots, \left(t_N, f(t_N)\right), 1 \leq n \ \leq N$ denote sampling coordinates of the time series. We define the labeling strategy $\mathcal{L}$ as a map:

$$\mathcal{L}: \left(t_n, f(t_n)\right) \to (t_n, C_n),$$

where C_n is a label in the set (Emergence, Growth, Saturation, Decline).

For a given $i \in \mathbb{N}$, we consider the set of points that satisfy:

$$\frac{d^i f}{dt^i} = 0, t > 0,$$

and define the inflection point $t_*^{(i)}$as the smallest point in this set. As a consequence, the stagnation point can be denoted as $f(t_*^{(1)})$. In other words, $t_*^1$is the point where the slope of the curve is zero. $t_*^2, t_*^3 \cdots$ are the points where the second, third etc. derivatives of the curve are zero. Note that, by construction, the stagnation point implicitly provides classification of a point on the maturity curve as either in (emergence, growth) phase or in (saturation, decline) phase.

We now propose a strategy to label phases in each subgroup. For ease of explanation, we only consider part of the curve up to the stagnation point and describe the proposed approach for labeling points in either emergence or growth phase. A similar approach can be used to label saturation and decline phases. Let us define A_t as area under the ideal maturity curve such that

$$A_t = \int_0^t f(t) dt$$

and the transition coefficient $\gamma \in [0,1]$ as

$$\gamma_i = \frac{A_{t_*^{(i)}}}{A_{t_*^{(1)}}}.$$

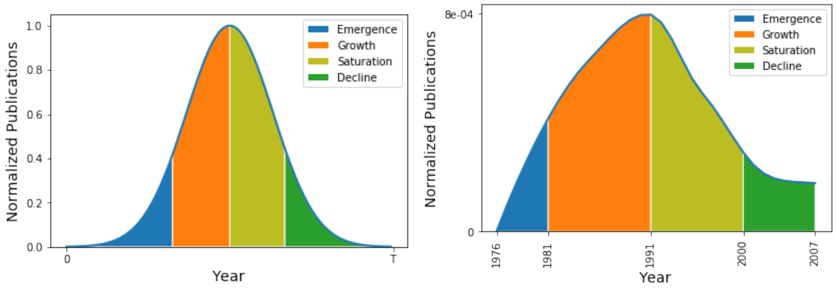
Here, γ_i is the measure of area up to an inflection point $t_*^{(i)}$ normalized w.r.t area up to the stagnation point $f(t_*^{(1)})$. For a choice of $i, c \in \mathbb{N}$, we define the following labeling rule

- $\frac{A_{t_n}}{A_{t_*^{(1)}}} \leq c\gamma_i, C_n = \text{Emergence}$
- $\frac{A_{t_n}}{A_{t_*^{(1)}}} > c\gamma_i, C_n = \text{Growth}$

Figure 5(a) shows labeling obtained with $i = 3$ and $c = 2$ which gives a good balance of distribution of data points between emergence and growth labels. Let us also define the transition point t_T such that C_n = Emergence for $t \leq t_T$, i.e. the point where the label transitions from emergence to growth.

Next, we generalize the labeling strategy to the maturity curve of any real technology. For doing so, we map the notion of transition point from the ideal curve to the maturity curve of a real technology. For particular choice of $i = 3$ and $c = 2$, we estimate numerically that $A_{t_T}/A_{t_*^{(1)}} \approx$ 0.2. We therefore use this metric to determine the transition point on the maturity curve of a real technology and label the points accordingly. Note that this mapping requires technologies in the training dataset to at least have the stagnation point for data points to have emergence and growth labels, if not the full technology life cycle. Since in this application we are mainly interested in technologies in either emergence or growth phases, training technologies with data

Commented [GS23]: I mostly follow this, but it's not quite clear to me what is driving that transition form emergence to growth, other than a point of sufficient momentum of publications having the stagnation point are sufficient. Figure 5(b) shows an example of this mapping from ideal maturity curve to the maturity curve for a real technology, optical computer in this case.



(a) (b)

Figure 5. Mapping of Emergence-Growth-Saturation-Decline onto curves. (a) An ideal curve. (b) An actual curve.

4 The Classification Problem

In this section, we describe the classification methodology developed in this study to determine the phase of evolution of a technology. We obtain data for classification by tracking 29 real technologies and labeling them with one of the four maturity phases, using the labeling method described in section 3.2. Data from 22 technologies were considered for training and validation (one of the technologies, "stable isotope analysis", did not have a stagnation point and was discarded) and the remaining 6 technologies were considered for testing the accuracy of the classifier. Figure 6(a) shows the distribution of data in each of the four labels.

The classification process consists of applying a hierarchy of binary classifiers. The first level of classification consists of using a binary classifier that distinguishes between data points in emergence/growth classes from data points in saturation/decline classes. Having obtained the first level of classification, we use a second binary classifier to distinguish data points between sub-phases i.e., emergence and growth for data points prior to stagnation point saturation and decline for the rest. Figure 7 gives a schematic representation of the classification procedure.

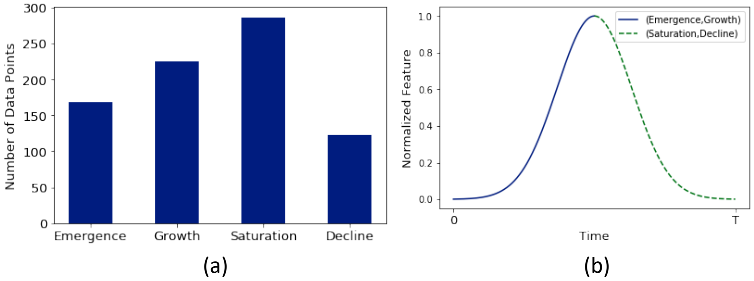


Figure 6: (a) Distribution of training data in four phases of technology maturity and (b) Region on ideal maturity curve belonging to (emergence, growth) and (saturation, decline) phases.

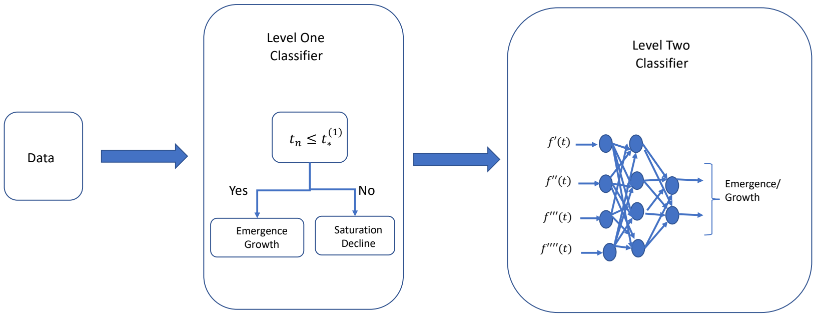


Figure 7: A schematic representation of the classification procedure.

4.1 Level One Classifier

The level one classifier is trivial and follows from the definition of stagnation point. We therefore have a simple rule-based classifier defined as follows:

If $t_*^{(1)}$ exists, then

- C_n = (Emergence or Growth), if $t_n \leq t_*^{(1)}$
- C_n = (Stagnation or Decline), $t_n > t_*^{(1)}$

Else

- C_n = (Emergence or Growth)

Figure 6(b) shows the application of this classifier on points of the ideal maturity curve. Since finding the existence and location of stagnation point is a simple deterministic procedure (calculation of $t_*^{(1)}$), the accuracy of this classifier is 100%.

Next, we propose a classifier to distinguish between sub-phases of a technology maturity curve.

4.2 Level Two Classifier

We consider the task of classification of data points in the emergence and growth phases and state that a similar approach can be used to distinguish between data points belonging to saturation and decline classes of the maturity cycle. Below, we list two important characteristics of the classification problem that informs our choice of features and the classification model.

- The training data comes from a time series and hence is sequential in nature.
- The shape of the maturity curve encodes the rate(s) of change in the observed variable, normalized publication count in this case, to track its evolution.

An ideal choice of classifier features and the classification model must naturally consider these specific characteristics of the problem. We note the following points on our choice of features and model.

- Numerical estimation of higher order derivatives (via finite-differences) at a given point on the maturity curve progressively takes into account the values of the maturity curve at adjacent locations. Thus, first and higher order derivatives of the maturity curve forms a promising set of features.

- Convolution Neural Networks naturally consider shape attributes and have been successfully used in image-based classification problems. If derivatives are used as features of the classification model, derivative stencils, in principle, can be considered as particular filter weights for the convolution operation.

We therefore choose first, second, third and fourth derivatives of $f(t)$ as classifier features and Neural Networks as the classification model.

Commented [GS24]: Nice description and conclusion

The neural net architecture (Goodfellow et al., 2016) considered in this work consists of an input layer with four nodes (one for each feature), one hidden layer in which the number of nodes is considered as a hyper-parameter and an output layer with two nodes, one for each class. We consider both mean squared error (MSE) and cross entropy (CE) loss functions and choose the one that gives an optimal model using Adam stochastic gradient descent algorithm (see Goodfellow et al., 2016 for details). The learning rate of the stochastic gradient descent algorithm is also considered as a hyperparameter. The output of the network is the probability of a data point belonging to the emergence class. Here, we use a standard threshold of 0.5 i.e., probability above 0.5 is considered as an emergence point. The threshold is a hyperparameter, and given that probabilities range from zero to one, 0.5 is the natural, naïve choice in the absence of hyperparameter tuning. The network was trained using the PyTorch framework (Paszke et al., 2019). In the following, we compare different classifier architectures based on the accuracy score (the fraction of examples in the training dataset that are correctly predicted by the classifier).

A sequential hyperparameter tuning procedure was used to select optimal architecture and learning algorithm parameters i.e., we first determined the optimal activation function, followed by the number of nodes in the hidden layer and finally the learning rate of the learning algorithm. The error for each hyperparameter combination was estimated on a separate set of data, the validation set, which was obtained by randomly selecting 20% of the training data before training the final model. Table 2 summarizes hyperparameter tuning results.

Based on hyperparameter tuning results shown in Table 2, we find that the optimal neural net architecture with an accuracy of 87.3% consists of the Tanh activation function with 5 hidden layer nodes and trained with MSE loss function with a learning rate of 5×10^{-2}. Using the selected network architecture and algorithm training parameters, we use the proposed classifier for classifying data points in the test technologies.

Table 2: Optimal hyperparameter selection based on the accuracy score for Mean Squared and Cross Entropy error measures. The hyperparameters considered are (a) activation function, (b) number of nodes in the hidden layer and (c) learning rate of the stochastic gradient descent algorithm.

Loss Function	**Activation Function**		
	Tanh	Sigmoid	Leaky ReLU
Mean Squared Loss	83.5%	77.2%	76.0%
Cross Entropy Loss	78.5%	81.0%	79.8%

(a)

Loss Function	**Nodes in Hidden Layer**		
	3	4	5
Mean Squared Loss	83.5%	86.1%	87.3%
Cross Entropy Loss	81.0%	83.5%	82.2%

(b)

Loss Function	**Learning Rate**	
	0.05	0.01
Mean Squared Loss	87.3%	84.8%
Cross Entropy Loss	83.5%	78.4%

(c)

4.3 Results on Test Technologies

Figure 8 shows classification of data points for test technologies in the emergence and growth classes using the level two neural net classifier.

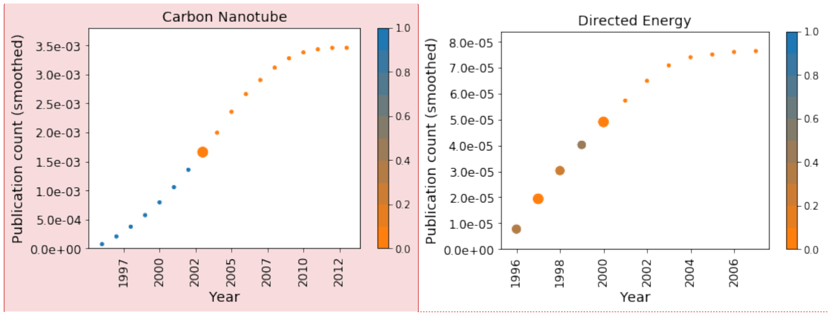


Figure 8: Classification of points on the maturity curve of two test technologies in emergence and growth classes using the level two classifier. The color of a point shows the probability of classifying a point in the emergence phase and its relative size indicates its deviation from the true value (i.e., 1 if the label is emergence and 0 otherwise).

The performance of the classifier on the six test technologies is in Table 3. The proposed classifier gives an average accuracy of 84.6% on data points on the test set which is close to the predicted accuracy of 87.3% of the optimal neural net architecture on the validation set. However, one limitation of this classifier is the wide range of accuracy scores on test technologies. For example, the classifier is 94.4% accurate for “carbon nanotube” but gives an accuracy score of 58.3% for “directed energy”. In order to overcome this limitation, we propose to generate more data to further improve the accuracy score of the classifier and to reduce its accuracy range.

Table 3: Accuracy scores on six test technologies

Technology	**Accuracy**
Microfluidics	87.5%
Carbon Nanotube	94.4%
Confocal Microscopy	84.2%
Atomic Force Microscopy	88.8%
Fuel Cells	94.1%
Directed Energy	58.3%

5 Data Augmentation and Refinements

Commented [GS25]: Maybe worth including the MSE or correct classification rate or something for these two technologies; it appears as though Directed energy is harder to correctly label based on the curve

Commented [GS26R25]: These values are summarized in the below table, it looks like this is the best and worst technology predictions

5.1 Generating Synthetic Data

The neural net classifier showed promise when it was trained on 22 technologies, achieving a 84.6% accuracy on average. There are two general approaches to improving the accuracy of any machine learning algorithm: hyperparameter optimization and increasing training set size. Refinement of the classifier via hyperparameter optimization is described in section 5.2. This section describes the methodology for generating synthetic data to bolster the accuracy of the neural net classifier. When a machine learning algorithm has more training data, it is able to learn about the entire range of cases it may encounter, thus improving its accuracy.

To create new, synthetic technology maturity time series, the following procedure was employed:

1. The 23 training technology maturity time series were re-discretized to each have 50 time points.
2. The 23 re-discretized time series were rescaled to be between zero and one, such that for any training technology i at time point t, the rescaled data point is given by
$$y^*_{i_t} = (y_{i_t} - min_i)/range_i$$
where y_{i_t} is the re-discretized data point, min_i is the minimum value of the curve for technology i, and $range_i$ is the difference between the maximum and minimum values of the curve for technology i.
3. Two training technologies were randomly sampled.
4. Two weights w_1 and w_2 were randomly sampled, such that
$$w_1 \sim \mho(0,\ 1)$$
where $\mho$ is the uniform distribution and
$$w_2 = 1 - w_1$$
5. The rescaled time series for the two randomly sampled training technologies were averaged using the randomly sampled weights, resulting in a new time series (Example in Figure 9(a)).
6. The new time series went through the reverse of the rescaling process in step two by applying the equation with a new minimum and range. The new minimum and range were sampled from their empirical distributions of the minimums and ranges of the 23 training technologies.
7. The time series was then re-discretized to a new number of time points. The new number of time points was randomly sampled from the density of the number of time points of the 23 training technologies, via inverse transform sampling. This results in a synthetic maturity time series to be used in the training set for the classifier (Example in Figure 9(b)).

Steps three through seven were repeated 500 times in order to produce 500 new, synthetic technology maturity curves. These synthetic data were used to train the classifier.

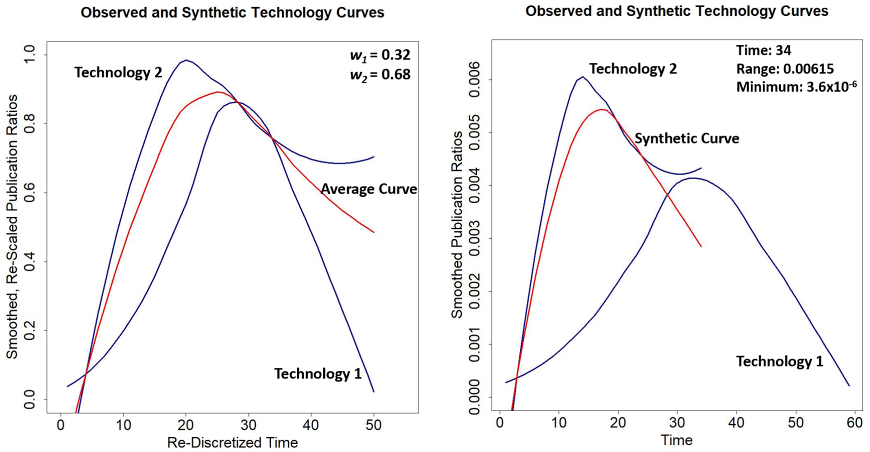


Figure 9(a). Training technologies 1 and 2 are re-discretized over time, rescaled, and then averaged to form an average curve. Figure 9(b). The average curve undergoes a reverse of the rescaling process and is re-discretized to create a synthetic training technology curve.

By expanding the size of the training set dramatically, the accuracy of the classifier improved (see Sec. 5.2). Leveraging synthetic data allowed the classifier to more easily learn about the distribution of the training data in low-dimensional space and hence a fuller range of possible curves it could encounter. This approach to generating synthetic data is not only effective in improving performance, it is also computationally inexpensive and simple to implement.

5.2 Level Two Classifier Refinement

In this section we use the 500 synthetic maturity curves as the training and validation data for new neural net classifier. The set of six test technologies that were used in Section 4.2 (and were not used to generate the synthetic data) are also used here to assess the performance of the new classifier. We choose the same architecture of the neural net with the exception of the number of nodes in the hidden layer. As we have more training data, we can have a higher number of nodes in the hidden layer. Here, we obtained the optimal value of 40 hidden nodes using hyperparameter tuning described in section 4.2. Figure 10 shows the convergence of mean squared error loss and accuracy score w.r.t the number of epochs in training the classifier. In addition, we optimize the probability threshold value for classification to the emergence class by doing a grid search in the set (0.4, 0.45, 0.5) and determine that the optimal value of 0.4 gives the smallest misclassification error for both emergence and growth classes.

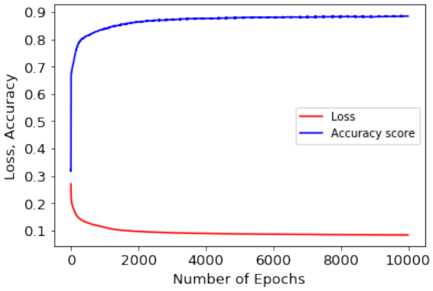


Figure 10: Convergence of mean squared error loss and accuracy score w.r.t the number of epochs in training the classifier

Table 4: Confusion matrix of the classifier trained on synthetic data. Misclassification error is 0.21 and 0.077 for emergence and growth classes respectively.

		Predicted	
		Emergence	Growth
Truth	Emergence	504	132
	Growth	83	988

Table 4 shows the confusion matrix obtained on the validation set with the classifier trained on synthetic data. Figure 11 shows classification of data points of the test technologies for which the classifier exhibited the best and the worst performance. In Table 5, we tabulate the accuracy score of the 6 technologies that were not used to construct the synthetic dataset. The average accuracy score on data points from the test set is 90.4% with the best accuracy of 100% (confocal microscopy) and the worst accuracy of 83.3% (directed energy). Thus, when optimized and trained on a dataset that was augmented with synthetic data, the classifier is more accurate overall, and its range of accuracy scores is narrower. Table A.1. shows the accuracy scores for all training and test technologies based upon the classifier trained with synthetic data.

Commented [GS27]: This was the description I had in mind for the previous one of these plots, not sure if I missed it or not

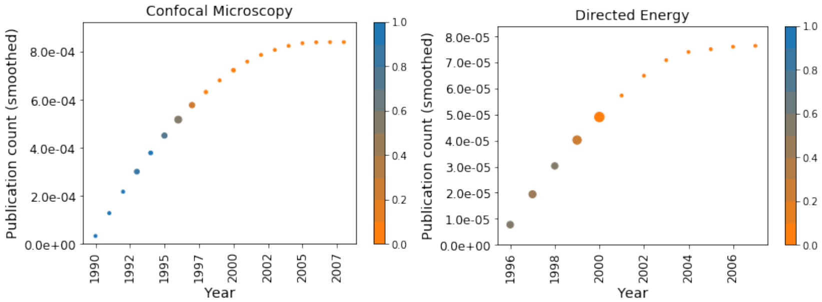


Figure 11: Classification of data points of (a) confocal microscopy (accuracy score = 100%) and (b) directed energy (accuracy = 83.3%) using the classifier trained on synthetic data. The color of a point shows the probability of classifying a point in emergence phase and its size indicates the deviation from the true value (i.e., 1 if the label is emergence and 0 otherwise). For confocal microscopy the points are small and there is a clear delineation in color between the two phases, indicating that the classifier predicts the boundary between emergence and growth clearly. For directed energy the points are larger, particularly near the boundary between emergence and growth, and the color is more neutral in the emergence phase. This indicates that the classifier struggles more to predict the boundary between emergence and growth, and that even when it does so successfully it is with a narrower margin.

Commented [GS28]: Much more thorough description than the same graphic in a previous section. Not necessary, but I'd be curious about why you think the Directed Energy maturation curve results in lower classification rates

Table 5: Accuracy scores on six test technologies obtained from classifier trained on synthetic data

Technology	Accuracy
Microfluidics	87.5%
Carbon Nanotube	88.9%
Confocal Microscopy	100%
Atomic Force Microscopy	88.9%
Fuel Cells	94.1%
Directed Energy	83.3%

6 Conclusions

We have investigated whether the technology life cycle curve, as quantified by peer-review journal / academic publications referring to the technology, can be used to predict the maturity of the technology. We are primarily interested in devising a way of identifying emergent technologies from open-source indicators, of which publications and patents are the most easily accessed, and which are also very closely linked to the actual development and evolution of the technology. Given this use-case, we focus on separating emergent technologies from growing ones. The premise behind our approach is that the time-series of publication counts, properly normalized, can yield features that are, in some sense, universal i.e., these features assume values that can be thought of as draws from distributions that are characteristic of the emergent and growth phases, and which are *not* affected by the identity of the technology. If the premise is true, it should be possible to construct a classification model, that given the features, can predict the maturity (emergent/growth) of an arbitrary technology.

Deleted: , a classifier

In this paper, we have shown that the premise may hold, and such a classifier is feasible. Using the time-series of annual publication counts (properly normalized) from 22 technologies, we compute the time-derivatives (that serve as features), learn a classifier and test its predictive skill on 6 held-out technologies. We obtain accuracies that range from 58.3% to 100% with an average accuracy of 84.6%. This performance was also helped by the fact that we developed a principled way of defining emergent and growth phases of a technology based on the fraction of publications that appear before the technology saturates. The absence of any labeling of data by experts spared us the complications of subjective bias. However, experts were used to develop the keyword searches to generate the technology curves, which could impart bias in the TLCs selected. The variability in the performance of our classifier is large and, we hypothesized, was mainly caused an unstable neural net classifier trained on too small a dataset. This numerical issue can be corrected by using a larger training dataset.

Commented [GS29]: May be worth including some additional data step descriptions, i.e. labeling of emergence vs growth in the training & testing data.

Commented [GS30R29]: It's the next sentence 😊

Commented [GS31]: Really nice paragraph!

Consequently, we augmented the dataset by randomly weighted averaging of the 23 original training technologies, selected two at a time. By doing so, we quickly created a large synthetic dataset on which we trained a new neural net classifier (a more stable one). Note that the classifier was trained solely on the synthetic data and then used to reclassify the 28 training and testing technologies. Our worst performing test case improved its accuracy score from 58.3% to 83%, a respectable performance. The performance of the classifier on the six held-out technologies varied between 100% and 83.3% with an average classification accuracy of 90.4%. In comparison, other modern machine learning approaches (which required labeled training data) are reported to have an average classification accuracy of 85.7% at maximum. In doing so, we demonstrated how a small dataset could be leveraged to reduce the errors in a classifier engendered by shortcomings of the training process. Note that the synthetic dataset does not contain entirely new information; it is derived from the original set of 23 technologies.

Commented [GS32]: Why reclassify the training data? I thought results were only reported for the testing data.

The ability to identify promising emergent technologies has its obvious uses in commerce, investment and the ability to maintain a competitive advantage. Current practices rely heavily on experts, which limits the number of technologies that can be assessed by them and is also affected by subjective bias. In contrast our method is free of experts with the exception of what search keywords to use when generating the TLCs, is purely algorithmic and consequently can be automated and scaled up. It raises the potential of constantly monitoring publication (and perhaps, patent) databases to unearth promising new developments in emergent fields of science and technology.

Commented [GS33]: Nice!! I really liked the paper overall, I think the only thing I'm still a little confused about is the phase labeling approach, other than that it all makes sense to me!

7 Acknowledgments

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

Effort sponsored in whole or in part by the Army Research Laboratory, U.S. Army, under Partnership Intermediary Agreement No. W911NF-15-3-0001. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Army Research Laboratory.

8 References

Achilladelis, B., & Antonakis, N. (2001). The dynamics of technological innovation: The case of the pharmaceutical industry. *Res Policy, 30*(4), 535-588.

Achilladelis, B., Schwarzkopf, A., & Cines, M. (1990). The dynamics of technological innovation: The case of the chemical industry. *Res Policy, 19*(1), 1-34.

Achilladelis, B. (1993). The dynamics of technological innovation: The sector of antibacterial medicines. *Res Policy,* 22(4), 279-308.

Albert, T., Moehrle, M.G., & Meyer, S. (2015). Technology maturity assessment based on blog analysis. *Technol Forecast and Soc Change, 92*, 196-209.

Albert, T. (2015). *Measuring technology maturity*. Springer Gabler.

Andersen, B. (1999). The hunt for S-shaped growth paths in technological innovation: A patent study. *J of Evol Econ, 9*, 487-526.

Ansoff, H.I. (1984). *Implanting strategic management*. Prentice-Hall.

Byun, J., Sung, T.E., & Park, H.W. (2018). Technological innovation strategy: How do technology life cycles change by technological area. *Technol Anal and Strat Manag, 30*(1), 98-112.

Dalkey, N., & Helmer, O. (1963). An experimental application of the DELPHI method to the use of experts. *Manag Sci, 9*(3), 351-515.

Ernst, H. (1997). The use of patent data for technological forecasting: The diffusion of CNC-technology in the machine tool industry. *Small Bus Econ, 9*, 361-381.

Ford, D., & Ryan, C. (1981). Taking technology to market. *Harvard Bus Rev, 59*(2), 117-126.

Franses, P.H. (1994). A method to select between Gompertz and logistic trend curves. *Technol Forecast and Soc Change, 46*(1), 45-49.

Gao, L., Porter, A.L., Wang, J., Fang, S., Zhang, X., Ma, T., Wang, W., & Huang, L. (2013). Technology life cycle analysis modeling based on patent documents. *Technol Forecast and Soc Change, 80*(3), 398-407.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. The MIT Press.

Haupt, R., Kloyer, M., & Lange, M. (2007). Patent indicators for the technology life cycle development. *Res Policy, 36*(3), 387-398.

Kim, B. (2003). Managing the transition of technology life cycle. *Technovation, 23*(5), 371-381.

Kim, J., Hwang, J., Jeong, D.H., & Jung, H. (2012). Technology trends analysis and forecasting application based on decision tree and statistical feature analysis. *Expert Syst with Appl, 39*(16), 12618-12625.

Lee, C., Kim, J., Kwon, O., & Woo, H.G. (2016). Stochastic technology life cycle analysis using multiple patent indicators. *Technol Forecast and Soc Change, 106*, 53-45.

Lemos, A.D., & Porto, A.C. (1998). Technological forecasting techniques and competitive intelligence: Tools for improving the innovation process. *Ind Manag and Data Syst, 8*(7), 330-337.

Lezama-Nicolás, R., Rodríguez-Salvador, M., Río-Belver, R., & Bildosola, I. (2018). A bibliometric method for assessing technological maturity: The case of additive manufacturing. *Scientometrics, 117,* 1425-1452.

Linden, A., & Fenn, J. (2003). Understanding Gartner's hype cycles. Gartner Inc.: Strategic Analysis Report.

Merino, D.N. (1990). Development of a technological S-curve for tire cord textiles. *Technol Forecast and Soc Change 37*(3), 275-291.

Momeni, A., & Rost, K. (2016). Identification and monitoring of possible disruptive technologies by patent-development paths and topic modeling. *Technol Forecast and Soc Change, 104*, 16-29.

Nagula, M. (2016). Forecasting of fuel cell technology in hybrid and electric vehicles using Gompertz growth curve. *J of Stat and Manag Syst, 19*(1), 73-88.

Nieto, M., Lozep, F., & Cruz, F. (1998). Performance analysis of technology using the S curve model: the case of digital signal processing (DSP) technologies. *Technovation, 18*(6-7), 439-457.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc. Retrieved from:

http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Ramadhan, M.H., Malik, V.I., & Sjafrizal, T. (2018). Artificial neural network approach for technology life cycle construction on patent data. In 2*018 5th International Conference on Industrial Engineering and Applications (ICIEA) IEEE*, pp. 499-503.

Ryu, J., & Byeon, S.C. (2011). Technology level evaluation methodology based on the technology. *Technol Forecast and Soc Change, 78*(6), 1049-1059.

Sharif, M.N., & Islam, M.N. (1980). The Weibull distribution as a general model for forecasting technological change. *Technol Forecast and Soc Change, 18*(3), 247-256.

Sick, N., Bröring, S., & Figgemeier, E. (2018). Start-ups as technology life cycle indicator for the early stage of application: An analysis of the battery value chain. *J of Cleaner Prod, 201*, 325-333.

Smojver, V., Štorga, M., & Potočki, E. (2019). Determining the life cycle phase of a technology based on patent data. *Tech Gaz, 26*(1), 222-229.

Sommerlatte, T., & Deschampes, J.P. (1986). Der strategische einsatz von technologien. In: Arthur D. Little International (Ed.), *Management im zeitalter der strategischen f**ü**hrung*. Gabler.

Stapleton, E. (1976). The Normal distribution as a model of technological substitution. *Technol Forecast and Soc Change, 8*(3), 325-334.

Su, H.N. (2017). How to analyze technology lifecycle from the perspective of patent characteristics? The cases of DVDs and hard drives. *R&D Manag, 43*(3), 308-319.

Taylor, M., & Taylor, A. (2012). The technology life cycle: conceptualization and managerial implications. *Int J of Prod Econ, 140*(1), 541-553.

Urban, G.L., & Hauser, J.R. (1993). *Design and marketing of new products*. Vol 2. Prentice hall.

van der Pol, J., & Rameshkoumar, J.P. (2017). The co-evolution of knowledge and collaboration networks: the role of the technology life cycle. *Scientometrics, 114*, 307-323.

Watts, R.J., & Porter, A.L. (1997). Innovation forecasting. *Technol Forecast and Soc Change, 56*(1), 25-47.

Wu, F.S., Hsu, C.C., Lee, P.C., & Su, H.N. (2011). A systematic approach for integrated trend analysis – The case of etching. *Technol Forecast and Soc Change, 78*(3), 386-407.

Appendix A.

Table A.1 Accuracy scores on training and test technologies obtained from classifier trained on synthetic data

Technology	Accuracy	Technology	Accuracy
Airbag	86.7%	Optical Computer	87.5%
High Temperature Superconductor	91.7%	Bubble Memory	88.9%
Scanning Tunneling Microscopy	100%	PCR	100%
Soft Lithography	90.9%	Green Fluorescent Protein	100%
Turbine Engine	94.6%	Ricin	100%
X-ray Lithography	94.7%	Photoresistor	90%
Gel Electrophoresis	96.4%	3G	100%
Patch Clamp	100%	4G	100%
MEMS	100%	Compact Disc	86.1%
Microfluidics	87.5%	Carbon Nanotube	88.9%
Confocal Microscopy	100%	Atomic Force Microscopy	88.9%
Fuel Cells	94.1%	Directed Energy	83.3%
GaAs	93.9%	Anthrax	NA*
YBaCuO	NA**	Cephalexin	NA**

* Anthrax had a rejuvenation which could not be classified as either emergence or growth. ** YBaCuO and cephalexin do not have any points in the emergence phase, and thus are not representative of the data expected during inference.

Vitae

Katherine Cauthen is a Senior Member of the Technical Staff at Sandia National Laboratories in the Complex Systems department. She obtained an M.S. in statistics at the University of New Mexico. Her research interests include anomaly detection, machine learning, and data fusion.

Prashant Rai is a Senior Data Scientist at Caterpillar Inc. He received his Ph.D. from the doctoral school of engineering sciences, Ecole Centrale Nantes, France. His research areas span uncertainty quantification, tensor methods for high dimensional functions, machine learning and their applications in science and engineering.

Nicholas Hale is a Senior R&D Specialist at Trillion Technology Solutions, Inc. His research is focused on applying machine learning methods to forecast and analyze science and technology. He obtained an M.S. in Computer Science at the Georgia Institute of Technology, a B.S. in Mathematics and a B.A. in Government from George Mason University.

Laura Freeman is a Research Associate Professor in Statistics and the Director of the Intelligent Systems Lab at the Virginia Tech Hume Center. Her research leverages experimental methods for conducting research that brings together cyber-physical systems, data science, artificial intelligence, and machine learning to address critical challenges in national security. Dr. Freeman has a B.S. in Aerospace Engineering, a M.S. in Statistics and a Ph.D. in Statistics, all from Virginia Tech.

Jaideep Ray Jaideep Ray is a Distinguished Member of the Technical Staff at Sandia National Laboratories, CA. He received his Ph.D. in Mechanical and Aerospace Engineering in 1999, from Rutgers, The State University of New Jersey. His interests lie in machine learning, Bayesian inference using scientific and engineering models and high-performance computing. He has contributed to compressible fluid dynamics and turbulence modeling. He has developed high-order methods for simulating reactive flows on block-structured adaptive meshes. He maintains and distributes open-source software. Details can be found at http://www.sandia.gov/~jairay