# Nutzung von Unsicherheit im teilüberwachten Lernen

Jonas Dauer
dauerjonas@googlemail.com
TU Dortmund
NRW, Germany

## ABSTRACT

Teilüberwachtes Lernen (SSL) befasst sich mit dem Lösen einer Klassifikations- oder Regressionsaufgabe mit Hilfe eines Datensatzes, der aus Daten besteht, die nur teilweise gelabelt sind. Das präsentierte Verfahren stellt eine Methode vor, die für ungelabelte Datenpunkte die Sicherheit der Vorhersage bestimmen soll und dafür sorgt, dass nur aus verlässlichen Vorhersagen gelernt wird. Das sogenannte *certainty-driven consistency loss*-Verfahren (*CCL*) basiert auf neuronalen Netzen und benutzt für die Vorhersage der Sicherheit der Klassifikation Augmentation und Dropout. Nachdem das Verfahren vorgestellt wurde, wird dieses mit aktuellen Verfahren, die ebenfalls für das teilüberwachte Lernen entwickelt wurden, verglichen.

**ACM Reference Format:**
Jonas Dauer. 2022. Nutzung von Unsicherheit im teilüberwachten Lernen. In *Proceedings of Seminar (Uncertainty quantification in machine learning)*. TU Dortmund, Germany, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 EINLEITUNG

Eine der aufwändigsten Arbeitsschritte für das Supervised Learning ist das Labeln der Daten. Laut Cognilytica nimmt das Labeln des Datensatzes circa 25% der Zeit, die ein überwachtes Lernprojekt benötigt, ein [3]. Es würde also die Kosten von vielen maschinellen Lernaufgaben erheblich senken, wenn es möglich wäre nur wenige gelabelte, dafür aber viele ungelabelte Daten zum Lernen nutzen zu können. Das teilüberwachte Lernen befasst sich mit dieser Aufgabenstellung. Fig. 1 zeigt den Nutzen ungelabelter Daten. Die links dargestellte Trennlinie der beiden Klassen, die im überwachten Lernen gelernt wurde, trennt beide gegebenen Klassen sehr gut. Nimmt man allerdings wie auf der rechten Seite weitere ungelabelte Datenpunkte hinzu, so findet man eine andere Trennlinie, die die beiden Klassen vermeintlich besser trennt. Mit dem Blick auf die Daten nehmen wir an, dass die ungelabelten Daten, die näher an den Kreisen sind auch als Kreise zu Klassifizieren sind. Dahingegen sollten die Datenpunkte, die näher an den Dreiecken sind auch als Dreiecke erkannt werden.

Die vorliegende Arbeit präsentiert das Paper *Certainty-Driven Consistency Loss for Semi-supervised Learning* [9]. Dieses stellt ein Verfahren vor, dass mit Hilfe von neuronalen Netzen das teilüberwachte Lernproblem zu lösen versucht. Es benutzt dazu zwei Netze. Zum einen ein Lehrer- und zum anderen ein Schüler-Netz. Das Verfahren schätzt dabei mit Hilfe von Dropout und Augmentation ab, wie sicher sich das Lehrer-Netz bei der Klassifikation eines gegebenen Datenpunktes ist und trainert das Schüler-Netz nur dann mit diesem Datenpunkt, wenn sich das Lehrer-Netz sicher genug ist, die richtige Klasse gefunden zu haben.

## 2 GRUNDLAGEN

Eine der grundlegenden Annahmen im teilüberwachten Lernen ist die *smoothness assumption* [1]. Die *smoothness assumption* behauptet, dass die Ausgaben zweier Punkte $x_1$ und $x_2$, die nahe beieinander liegen, ebenfalls nahe beieinander liegen sollen. Diese Aussage ist allgemein gefasst und z.B. auch für Regressionsaufgaben anwendbar. Für die Klassifikation kann man sie so interpretieren, dass die nahe beieinander liegenden Punkte $x_1$ und $x_2$ einer Klasse $y$ zugeordnet werden sollen. Mit der *smoothness assumption* kann die in Fig. 1 auf der rechten Seite gefundene Trennlinie mathematisch begründet werden.

Wie bereits erwähnt werden für die Abschätzung der Unsicherheit des Lehrer-Netzes Dropout und Augmentation als Techniken eingesetzt. Beim Dropout-Verfahren [14] werden dabei die Ausgaben zufälliger Neuronen einer versteckten Schicht im neuronalen Netz ignoriert, also mit 0 multipliziert. Dies führt dazu, dass aus dem gesamten Netz zufällige Subnetze extrahiert werden. Diese Subnetze haben die Eigenschaft, dass sie für die gleiche Eingabe unterschiedliche Ausgaben erzeugen. Die Autoren des vorgestellten Verfahrens nehmen nun an, dass die Unsicherheit der Vorhersage des gesamten Netzes mit den Unterschieden in den Verhersagen der erzeugten Subnetze korreliert. Je größer die Unterschiede



*Uncertainty quantification in machine learning, SS 2022, Dortmund, NRW, Germany*



ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$0.00
https://doi.org/XXXXXXX.XXXXXXX

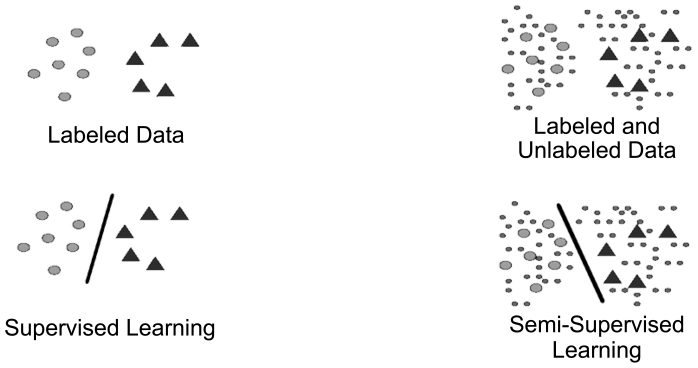


**Figure 1: Beispiel für den Nutzen ungelabelter Daten. Links: Supervised Learning. Rechts: Semi-Supervised Learning [11].**

in den Vorhersagen, desto unsicherer ist sich das gesamte Netz.
Das zweite eingesetzte Verfahren für die Bestimmung der Unsicherheit der Klassifikation ist die Augmentation. Das Ziel der Augmentation ist es eine gelabelte Eingabe möglichst viel zu verändern, ohne allerdings ihr Label zu ändern. Beispiele einer Augmentation für Bilddaten ist die Translation, Rotation oder ein Zoomen. Dabei entstehen neue Datenpunkte, die im Eingaberaum in der Nähe zu den originalen Eingaben liegen. Um die Unsicherheit der Klassifikation zu bestimmen wird die Differenz der unterschiedlichen Durchläufe der Klassifikationen gemessen. Je größer die Unterschiede für die Klassifikation augmentierter Daten, desto unsicherer ist die Entscheidung. Fig. 2 zeigt auf der linken Seite die Bestimmung der Unsicherheit einer Klassifizierung mit Hilfe des Dropout-Verfahrens und auf der rechten Seite die Bestimmund der Unsicherheit mit Hilfe des Augmentation-Verfahrens.

## 3 ARCHITEKTUR

Bevor ein detaillierterer Blick auf die Funktionsweise des Verfahrens geworfen wird, soll erst die genutzte Architektur vorgestellt werden. Wie bei jeder teilüberwachten Lernaufgabe besteht der Datensatz $D$ aus einem kleinen Teil gelabelter $D_L$ und aus einem großen Teil ungelabelter Daten $D_U$. Mit Hilfe der gelabelten Daten $D_L$ ist es möglich ein neuronales Netz überwacht zu trainieren. Dieses Netz wird als Lehrer-Netz bezeichnet und hat die Gewichte $\theta'$. Mit Hilfe des Lehrer-Netzes wird im nächsten Schritt das sogenannte Schüler-Netz trainiert. Das Schüler-Netz hat dabei die selbe Architektur wie das Lehrer-Netz. Mit Hilfe des Dropout- und des Augmentation-Verfahrens wird bestimmt, wie unsicher das Lehrer-Netz die Klassifikation eines Datenpunktes vornimmt. Das genaue Vorgehen zur Bestimmung der Unsicherheit wird in Abschnitt 4 vorgestellt. Ist die Klassifikation des Lehrer-Netzes für einen Datenpunkt relativ sicher (was "relativ sicher" bedeutet wird ebenfalls in Abschnitt 4 erläutert), wird das Schüler-Netz (Gewichte: $\theta$) mit diesem trainiert. Dabei ist es das Ziel für einen gegebenen Batch $B$ mit $|B|$ Trainingsbeispielen auf der einen Seite den überwachte Loss für die gelabelten Daten und auf der anderen Seite den *consistency* Loss für alle Daten zu minimieren. Der *consistency* Loss beschreibt den Unterschied in der Vorhersage des Lehrer- und des Schüler-Netzes. Wie in Fig. 4 dargestellt wird Wissen, das das Schüler-Netz durch das Trainieren mit ungelabelten Daten gewonnen hat, mit Hilfe eines *exponential moving average* (EMA) [15] in das Lehrer-Netz zurückgeführt. Der EMA verschiebt die Gewichte des Lehrer-Netzes dabei in die Richtung der korrespondierenden Gewichte des Schüler-Netzes und wird wie folgt berechnet:

$$\theta'_s = \alpha\theta'_{s-1} + (1-\alpha)\theta_s.$$

Der Faktor $\alpha$ ist dafür zuständig die Geschwindigkeit, mit der die Information aus dem Schüler- in das Lehrer-Netz fließt, zu kontrollieren und liegt zwischen 0 und 1. Ein kleines $\alpha$ bedeutet, dass die Information schnell fließt, während sich bei einem großen $\alpha$ das Lehrer-Netz nur langsam dem Schüler-Netz annähert. Das Lehrer-Netz kann das aus den ungelabelten Daten gewonnene Wissen also ebenfalls nutzen um bessere Vorhersagen zu treffen. Es entsteht ein Kreislauf, in dem sich das Lehrer- und das Schüler-Netz gegenseitig ständig verbessern. Um zu verhindern, dass vor allem zu Beginn des Trainings Informationen aus dem wahrscheinlich noch nicht gut trainierten Schüler-Netz in das Lehrer-Netz zurückfließen, wird für gewöhnlich ein größeres $\alpha$ (z.B. 0,99) gewählt [15]. Außerdem würde ein kleineres $\alpha$ dafür sorgen, dass sich die Gewichte des Lehrer-Netzes schnell ändern, was dafür sorgen würde, dass das Lernen instabil werden würde. Zumindest lassen sich vergleichbare Effekte im Reinforcement Learning finden [5]. Die Architektur des Verfahrens ist in Fig. 4 dargestellt.

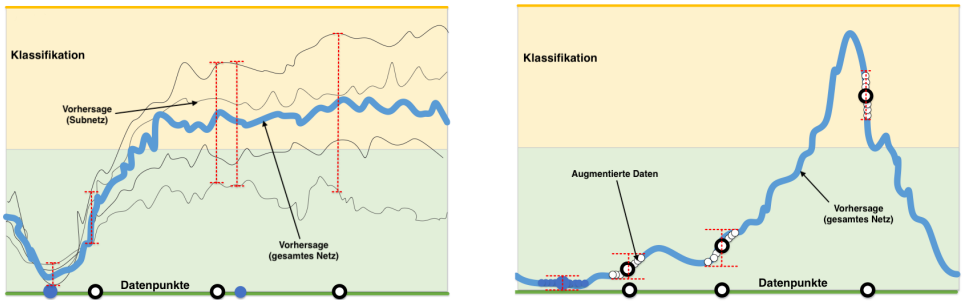


**Figure 2: Dropout und Augmentation im Eingaberaum. Auf der x-Achse sind die Datenpunkte und auf der y-Achse das Ergebnis der Klassifikation abgebildet. Links: Das Dropout-Verfahren erzeugt mehrere Subnetze, die alle eine eigene Vorhersagekurve erzeugen. Je größer die Differenz der Vorhersage der einzelnen Subnetze, desto unsicherer ist das gesamte Netz. Rechts: Das Augmentation-Verfahren erzeugt aus einer gegebenen Eingabe weitere Eingaben, die aber nahe an der ursprünglichen liegen. Je kleiner die Differenz aller dieser Vorhersage ist, desto sicherer ist sich das Netz mit der Klassifikation.**

Von besonderer Bedeutung ist weiterhin die Loss-Funktion des Schüler-Netzes. Während das Lehrer-Netz klassisch überwacht trainiert wird (z.B. mit der Kreuzentropie), kommt für das Schüler-Netz noch ein Term für den sogenannten *consistency Loss* hinzu. Dieser *consistency Loss* gibt den Unterschied der beiden Vorhersagen von dem Lehrer- und dem Schüler-Netz an und soll ebenfalls minimiert werden. Die gesamte Loss-Funktion sieht dann wie folgt aus:

$$\min_{\theta} \sum_{x_i \in (B \cap D_L)} L_{cls}(f(x_i, \theta, \eta), y_i) + \lambda(e) \sum_{x_i \in B} L_{cons}(f(x_i, \theta', \eta') - f(x_i, \theta, \eta)).$$

Die Funktion $L_{cls}$ ist die Standard Kreuzentropy, während $L_{cons}$ den Mittleren Quadratischen Fehler (MSE) der beiden Vorhersagen des Lehrer- und des Schüler-Netzes bestimmt. Der Faktor $\lambda(e)$ hängt von der Trainingsepisode $e$ ab und beschreibt die Abwägung zwischen dem überwachten und dem *consistency* Loss. Dieser Faktor ist zunächst relativ klein und sorgt somit dafür, dass der überwachte Loss beforzugt minimiert wird. Im Laufe des Trainings allerdings wird $\lambda(e)$ größer, wodurch der *consistency* Loss immer wichtiger wird.

## 4 INTEGRATION DER UNSICHERHEIT

In diesem Abschnitt wird aufgezeigt werden, wie Certainty-Driven Consistency Loss (CCL) die Unsicherheit der Klassifikation eines neuronalen Netzes bestimmt und wann eine Klassifikation des Lehrer-Netzes als sicher genug gilt, damit das Schüler-Netz trainiert werden kann. Als erstes zieht man aus den Daten $B$ (gelabelte und ungelabelte Datenpunkte) einen Batch $B'$. Aus diesem Batch $B'$ bestimmt man dann

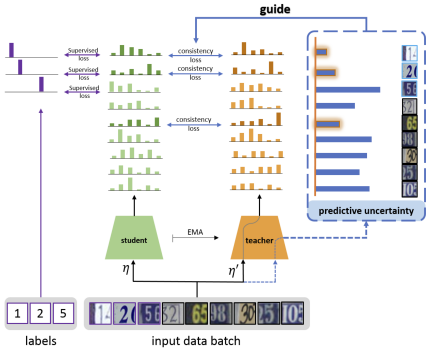


**Figure 3: Architektur des CCL mit Filterung. Das Schüler-Netz wird nur mit den Datenpunkten trainiert, für die die Klassifikation des Lehrer-Netzes als sicher gilt (consistency loss).**

für jeden Datenpunkt $x_i$ $T$ mal (z.b. $T = 20$) unter zufälligem Dropout und zufälliger Augmentation den softmax Wahrscheinlichkeitsvektor

$$[p(y = 1|x, \hat{\theta'^t}, \hat{\eta'^t}), ..., p(y = C|x, \hat{\theta'^t}, \hat{\eta'^t})],$$

wobei $\hat{\theta'^t}$ den random Dropout und $\hat{\eta'^t}$ die Augmentation der Eingaben beschreibt. Mit Hilfe dieser Vektoren lassen sich nun verschiedene Kriterien definieren, die als Maß für die Unsicherheit der Klassifizierung genutzt werden. Dabei sollen die Kriterien folgende Bedingungen erfüllen. (1) Damit das Kriterium überhaupt angewendet werden kann muss es in der Lage sein die Varianz über $T$ Vektoren zu bestimmen. (2) Das Kriterium muss die Wahrscheinlichkeitsverteilung der Vektoren in Betracht ziehen. Um die Unsicherheit der Klassifikation zu bestimmen soll das Kriterium z.B. nicht zählen wie oft eine Klasse als die Wahrscheinlichste bestimmt

wurde, sondern die Verteilung der Wahrscheinlichkeiten aller Klassen berücksichtigen. (3) Die Ausgabe des Kriteriums ist ein kontinuierlicher Skalar. Es ist damit sichergestellt, dass die Unsicherheiten verschiedener Datenpunkte leicht miteinander verglichen werden können.
In dem Paper werden insgesamt 4 Kriterien vorgestellt, die die genannten Bedingungen erfüllen. Es soll an dieser Stelle exemplarisch die *predictive variance* (PV) vorgestellt werden. Die PV summiert die Varianz der $T$ zufälligen Vorhersagen für alle Klasse auf. Je größer die Ausgabe des Kriteriums, desto größer ist die Unsicherheit der Vorhersage. Die PV ist wie folgt definiert:

$$PV = \sum_{c} Var[p(y = c|x, \hat{\theta'^1}, \hat{\eta'^1}), ..., p(y = c|x, \hat{\theta'^T}, \hat{\eta'^T})].$$

Die weiteren Kriterien sind die *entropy variance* (EV), die *predictive entropy* (PE) und die *multual information* (MI). Wie die Namen schon erahnen lassen basieren diese Methoden zum Teil auf der Summe der Entropien der Wahrscheinlichkeiten, dass das gegebene Muster zu einer gegebenen Klasse gehört. Für den Rahmen dieser Arbeit reicht die genauere Kenntnis über die PV allerdings aus. Die genauen Definitionen der anderen Kriterien können in [9] nachgeschlagen werden.
Im Folgenden werden zwei Verfahren vorgestellt, die dafür sorgen, dass das Schüler-Netz nur mit Trainingsdaten trainiert wird, die von dem Lehrer-Netz als sicher klassifiziert worden sind.

## 4.1 *Certainty-driven consistency* mit Filterung

Die Grundlegende Idee des *Certainty-driven consistency* mit Filterung ist es nur von den Daten zu lernen, die sicher klassifiziert wurden. Dazu wird als erstes ein Batch $B'$ mit $|B'|$ vielen Daten aus den Trainingsdaten gezogen. Als nächstes berechnet der Algorithmus für alle diese Daten des Batches das Kriterium, z.B. die *predictive variance*. Im nächsten Schritt werden alle Daten aufsteigend nach der Unsicherheit ihrer Klassifikation geordnet. Die Daten, die am sichersten Klassifiziert wurden, haben den niedrigsten Rang. Die Daten, deren Klassifikation durch das Lehrer-Netz als unsicherer eingeschätzt wurden, haben den höchsten Rang. Aus diesen geordneten Datenpunkten werden nun die $k$ ersten ausgewählt. Dies sind die Datenpunkte, für die sich das Lehrer-Netz am sichersten mit der Klassifikation war. Wir bezeichnen diese ausgewählten Daten als $B_k$. Dieses Vorgehen alleine würde sehr wahrscheinlich dafür sorgen, dass sich das Schüler-Netz auf die sichersten Daten überanpasst, da es lediglich mit diesen trainiert werden würde. Aus diesem Grund findet noch eine zufallsbasierte Filterung statt. Diese filtert per Zufall eine Menge an Trainingsdaten aus $B_k$ heraus. Allerdings hängt auch diese Filterung von der Unsicherheit der Klassifikation durch das Lehrer-Netz ab. So werden ebenfalls sichere Datenpunkte mit einer höheren Wahrscheinlichkeit für das Trainieren des Schüler-Netzes ausgewählt.
Um aber den *consistency loss* zu minimieren fehlt noch ein Schritt. Der *consistency loss* beschreibt den Unterschied zwischen dem Lehrer- und dem Schüler-Netz. Wenn $k$ konstant bleibt und kleiner ist als die Anzahl der Trainingsdaten ($|B|$), dann wird das Schüler-Netz nicht mit den Trainingsdaten trainiert, bei denen sich das Lehrer-Netz unsicher ist. Um allerdings die Konsistenz der beiden Netze zu erreichen muss das Schüler-Netz auch mit diesen Daten trainiert werden. Aus diesem Grund wird der Parameter $k$ mit jeder Iteration ein wenig erhöht, bis dieser den Wert $|B|$ erreicht hat und die Möglichkeit besteht, dass das Schüler-Netz mit allen Trainingsdaten trainiert wird.
Da im Verlauf des Trainings die Unsicherheit, mit der das Lehrer-Netz Vorhersagen trifft kleiner werden sollte, sollte das Schüler-Netz außerdem die Möglichkeit haben mit mehr Trainingsdaten trainiert zu werden. Auch dies wird mit der Erhöhung von $k$ erreicht.
Beim *Certainty-driven consistency* mit Filterung wird das Schüler-Netz also nur mit den Trainingsdaten trainiert, für die sich das Lehrer-Netz am sichersten ist. Zu beachten ist, dass die $k$ Trainingsdaten, mit denen das Schüler-Netz trainiert wird, sowohl aus den ungelabelten, wie auch aus den gelabelten Daten gezogen werden.

## 4.2 *Certainty-driven consistency* mit Temperatur

Die Autoren des Papers stellen eine zweite Variante ihres Ansatzes vor. Bei der *Certainty-driven consistency* mit Temperatur verfolgen sie einen anderen Ansatz. Anstatt Datenpunkte, deren Klassifikation als unsicher eingeschätzt werden, nicht für das Training zu benutzen, reguliert der Ansatz mit Temperatur wie viel aus diesen Beispielen gelernt wird. Ein Grund, weshalb man auch aus unsicheren Daten lernen sollte, ist z.B. um die Ähnlichkeit zweier Klassen zu bestimmen. Ist sich das Lehrer-Netz z.B. sicher, dass ein Datenpunkt zu einer von zwei Klassen gehört, liefert aber kein sicheres Ergebnis zu welcher der beiden Klassen der Datenpunkt gehört, so kann man daraus schließen, dass sich die beiden Klassen in bestimmten Bereichen des Eingaberaums ähneln. Informationen über die Ähnlichkeit zwei Klassen werden als "dunkles Wissen" bezeichnet [6]. Zum Teil wird das "dunkle Wissen" bereits im ersten Ansatz genutzt, aber in der *Certainty-driven consistency* mit Temperatur wird dieses Wissen häufiger verwendet.
Die Temperatur wird in der *softmax*-Schicht angewandt.

Dabei sieht die *softmax*-Funktion wie folgt aus:

$$q_i = \frac{exp(z_i/V)}{\sum_j exp(z_j/V)}.$$

Dabei bezeichnet $z_i$ die Ausgabe des $i$-ten Neurons der Ausgabeschicht des Lehrer-Netzes und $V$ die Temperatur. Für $V = 1$ erhält man die standard *softmax*-Funktion. Wenn $V$ allerdings größer wird, dann gleicht sich die *softmax*-Funktion immer mehr einer Gleichverteilung an. Dadurch wird die Verteilung, die von dem Lehrer-Netz vorgegeben wird, je nach Größe von $V$ in Richtung einer Gleichverteilung abgeändert und sorgt dafür, dass der wahrscheinlichsten Klasse nicht eine überproportional große Bedeutung zugeschrieben wird.
Ebenso wie bei dem Ansatz mit der Filterung wird auch bei dem *Certainty-driven consistency* mit Temperatur zunächst ein Minibatch $B'$ aus den Trainingsdaten gezogen und dieser aufsteigend nach der Unsicherheit der Klassifikation sortiert. Mit Hilfe eines maximalen Wertes $V_{max}$ für $V$ und dem Rang der Unsicherheit des Trainingsdatenpunktes wird $V$ wie folgt berechnet:

$$V = \frac{V_{max}}{|B'|} R_i.$$

Dabei wird im Laufe des Trainings durch die stetige Verkleinerung von $V_{max}$ dafür gesorgt, dass der bereits erwähnte *consistency loss* minimiert wird, da sich die Verteilung, mit der das Schüler-Netz trainiert wird, immer weiter der Ausgabeverteilung des Lehrer-Netzes annähert.

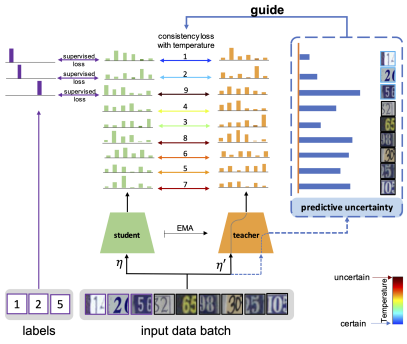


**Figure 4: Architektur des CCL mit Temperatur. Das Schüler-Netz wird mit allen Datenpunkte trainiert. Allerdings bestimmt der Rang der Unsicherheit des Datenpunktes wie hoch die Temperatur ist und wie viel das Schüler-Netz aus den gegebenen Trainingsdaten lernen soll [9].**

# 5 EXPERIMENTE

Die Autoren des Papers haben ihr Verfahren auf den Datensätzen CIFAR-10 [7], SVHN [13] und CIFAR-100 [7] getestet und einen Vergleich zu anderen modernen Methoden gezogen. Dabei wurden von allen gelabelten Datenpunkten zufällig bei jeweils 1.000, 2.000, 4.000 oder 10.000 das Label behalten, während die anderen Daten ungelabelt genutzt wurden. Es wurde die selbe 10-Schichten CNN Netzwerk-Architektur wie in [15] und [8] für die Experimente genutzt. Die Ergebnisse sind in Fig. 5 abgebildet. Für die Experimente wurden jeweils 10 Durchläufe durchgeführt, in denen jeweils die gelabelten Daten zufällig ausgewählt wurden. Die *supervised-only* Methode wurde jeweils nur mit den gelabelten Daten trainiert. Auf den ersten Blick ist ersichtlich, dass diese Methode auch die schlechtesten Ergebnisse liefert. Es wurde also bestätigt, dass ungelabelte Daten das Ergebnisse einer Klassifikation verbessern.
Bezüglich der beiden vorgestellten Verfahren, *Filtering CCL* und *Temperature CCL*, fällt auf, dass sie entweder besser abschneiden als die existierenden Methoden, oder aber diesen in ihrer Fehlklassifikationsrate sehr nahe kommen. Besonders bei dem CIFAR-10 Datensatz mit nur 1.000 oder 2.000 gelabelten Daten schneiden die beiden vorgestellten Ansätze im Vergleich zu den anderen Methoden sehr gut ab.

## 5.1 Genauigkeit vs. Unsicherheit

Ebenfalls interessant ist der Vergleich zwischen der Genauigkeit der Klassifikation und der geschätzten Unsicherheit. Fig. 6 zeigt dieses Verhältnis auf. Hier wurde mit Hilfe des CIFAR-10 Datensatzes ein Schüler-Netz trainiert, das nach dem Training ausgewertet wurde. Die Menge $D_T$ beschreibt dabei die Testmenge. Die Genauigkeit wurde dabei über die jeweilige Klasse gemittelt. Es werden also z.B. circa 95% aller Automobile auch als solche erkannt. Ebenfalls über eine Klasse gemittelt wurde die *predictive variance*. Fig. 6 zeigt, dass eine Klasse im Allgemeinen besser erkannt wird, wenn sich das Netz, welches zur Klassifikation genutzt wurde, sicherer gemäß des PV-Kriteriums für diese Klasse ist. Dieser Zusammenhang unterstützt die Annahme, dass die *predictive variance* als Maß für die Unsicherheit der Klassifikation dienen kann.
Der gleiche Zusammenhang lässt sich auch bei den anderen im Paper vorgestellten Kriterien beobachten. Fig. 7 zeigt den antiproportionalen Zusammenhang zwischen der Genauigkeit und der Unsicherheit der Klassifizierung auf. Es wurde der CIFAR-10 Datensatz mit 1.000 gelabelten und 49.000 ungelabelten Daten genutzt. Für alle ungelablten Daten wurde mit Hilfe des Dropout-Verfahrens in 20 zufälligen Durchläufen die Unsicherheit anhand der 5 Kriterien PV, EV, PE, MI und *predictive ratio* (PR) bestimmt. PR ist dabei

| Model | CIFAR-10 1000 labels | CIFAR-10 2000 labels | CIFAR-10 4000 labels | SVHN 1000 labels | CIFAR-100 10000 labels |
|---|---|---|---|---|---|
| Supervised-only | 46.43 ± 1.21 | 33.94 ± 0.73 | 20.66 ± 0.57 | 12.32 ± 0.95 | 44.56 ± 0.30 |
| Π model | – | – | 12.36 ± 0.31 | 4.82 ± 0.17 | 39.19 ± 0.36 |
| TempEns | – | – | 12.16 ± 0.24 | 4.42 ± 0.16 | 38.65 ± 0.51 |
| VAT+Ent | – | – | 10.55 ± 0.05 | 3.86 ± 0.11 | – |
| MT | 21.55 ± 1.48 | 15.73 ± 0.31 | 12.31 ± 0.28 | 3.95 ± 0.19 | 37.91 ± 0.37 |
| Π+SNTG | 21.23 ± 1.27 | 14.65 ± 0.31 | 11.00 ± 0.13 | **3.82 ± 0.25** | 37.97 ± 0.29 |
| MT+SNTG | – | – | – | 3.86 ± 0.27 | – |
| TempEns+SNTG | 18.41 ± 0.52 | 13.64 ± 0.32 | 10.93 ± 0.14 | 3.98 ± 0.21 | – |
| MA-DNN | – | – | 11.91 ± 0.22 | 4.21 ± 0.12 | **34.51 ± 0.61** |
| Filtering CCL (**ours**) | **16.99 ± 0.71** | 12.57 ± 0.47 | **10.63 ± 0.22** | 3.86 ± 0.19 | 34.81 ± 0.52 |
| Temperature CCL (**ours**) | 17.26 ± 0.69 | **12.45 ± 0.33** | 10.73 ± 0.26 | 3.93 ± 0.21 | 35.15 ± 0.62 |

**Figure 5: Resultate des CCL-Verfahrens auf den Datensätzen CIFAR-10, SVHN und CIFAR-100 im Vergleich mit anderen modernen Methoden. Die Ergebnisse wurden über 10 zufällige Durchläufe gemittelt und stellen die Fehlerrate (%) dar. Als zweites ist hinter dem Plusminuszeichen (±) die Standardabweichung angegeben. Die angegebenen Verfahren lassen sich wie folgt finden: Supervised-only [15], Π model [8], TempEns [8], VAT+Ent [12], MT [15], Π+SNTG [10], MT+SNTG [10], TempEns+SNTG [10], MA-DNN [2].**

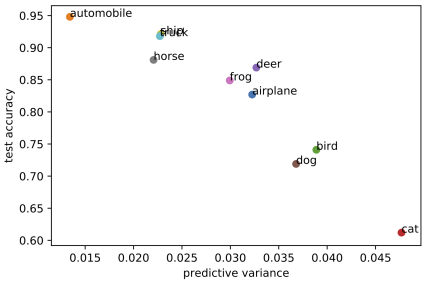


**Figure 6: Genauigkeit vs. PV auf $D_T$**

ein Kriterium, welches nicht den drei festgelegten Bedingungen entspricht, weil es nur die am als wahrscheinlichsten vorhergesagte Klasse betrachtet. Nachdem die Unsicherheit bestimmt wurde, wurden die 49.000 Datenpunkte aus der Menge $D_U$ entsprechend ihrer Unsicherheit aufsteigend sortiert. Im Anschluss an die Sortierung wurden je 100 Bilder zu sogenannten "bins" zusammengefasst. Die Fig. 7 zeigt also 490 "bins" zu je 100 Bildern, die nach ihrer Unsicherheit aufsteigend sortiert sind, auf der x-Achse. Auf der y-Achse ist die Genauigkeit der jeweiligen Klassifikation abgebildet. Wie bereits bei dem Kriterium PV gesehen lässt sich auch für die anderen Kriterien, die den drei aufgestellten Bedingungen entsprechen ein antiproportionaler Zusammenhang zwischen der Genauigkeit der Klassifikation und der Unsicherheit dieser beobachten. Lediglich das PR-Kriterium weist einige Abweichungen von dieser These auf. Dies kann als Hinweis darauf gedeutet werden, dass die aufgestellte zweite Bedingung, die behauptet, dass ein Kriterium die Wahrscheinlichkeitsverteilung aller Klassen und nicht nur die Klasse mit der größten Wahrscheinlichkeit zu betrachten hat, ihre Berechtigung hat. Es scheint so, als ist die PR kein besonders gutes Maß für die Unsicherheit der Klassifizierung. Allerdings wird für diese These keine Begründung geliefert. Auch die Quelle aus der die Autoren die *predictive ratio* entnommen haben [4] liefert dazu keine Erkenntnisse. Eine Vermutung ist, dass die softmax-Aktivierung dafür sorgt, dass die Wahrscheinlichkeitsverteilung der Klassen sehr stark in Richtung der favorisierten Klasse verschoben wird. Dadurch ist es sehr unwahrscheinlich, dass ein Datenpunkt unter Dropout oder Augmentation in mehreren Durchläufen zu verschiedenen Klasse zugeordnet wird. Bei einer Mehrklassenentscheidung müsste die Wahrscheinlichkeit für die Zugehörigkeit zu einer bestimmten Klasse deutlich unter 50% sinken, damit dieser Punkt einer anderen Klasse zugeordnet wird. Die Ungenauigkeiten in der Klassifikation müssten also deutlich größer sein, damit das *predictive ratio* sinkt.

## 5.2 Verrauschte Daten

In Fig. 8 wurden die drei Verfahren überwachtes Lernen, MT [15] und CCL mit Filterung mit verrauschten Labeln trainiert. Dazu wurden in dem CIFAR-10 Datensatz mit 4.000

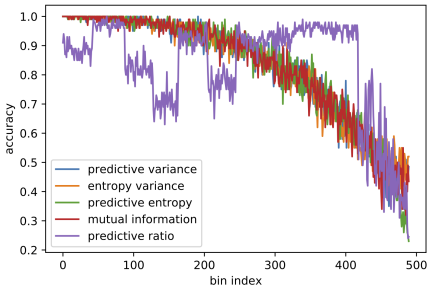


**Figure 7: Antiproportionaler Zusammenhang zwischen Genauigkeit und Unsicherheit der Klassifizierung anhand des CIFAR-10 Datensatzen, der mit 1.000 gelabelten und mit 49.000 ungelabelten Daten trainiert wurde. Ein "bin" enthält 100 Bilder. Bis auf *predictive ratio* ist der antiproportionale Zusammenhang überall zu beobachten.**

gelabtelten Daten je 20%, 30% oder 50% der Labels zufällig manipuliert. Die Graphen zeigen den Klassifikationsfehler der jeweiligen Verfahren. Es ist gut zu erkennen, dass sich der überwachte Lernansatz vor allem bei den 30% und den 50% veränderten Labels stark überanpasst, was dem erwarteten Verhalten entspricht. Besonders interessant ist aber die Robustheit der beiden anderen Verfahren gegenüber dem Rauschen. Wenn man den durchschnittlichen Klassifikationsfehler der letzten 20 Trainingsepochen zwischen MT [15] und dem CCL mit Filterung vergleicht, fällt auf, dass der vorgestellte CCL-Ansatz deutlich besser abschneidet. Bei der Manipulation von 20% der Labels (bei 4.000 gelabelten Daten sind dies 800 Datenpunkte) erreicht der CCL-Ansatz eine niedrigere Fehlklassifikationsrate von 7,26% (Fehlerrate MT: 27,14%; Fehlerrate CCL mit Filterung: 19,88%). Je mehr Labels verfälscht werden, umso kleiner wird allerdings der Unterschied der beiden Verfahren. CCL mit Filterung liefert allerdings für alle aufgezeigten Szenarien die besseren Ergebnisse. Bei 30% manipulierter Labels performt CCL mit Filterung 6,27% besser als MT [15] (Fehlerrate MT: 35,79%; Fehlerrate CCL mit Filterung: 29,52%). Auch bei den 50% verrauschten Labeln schneidet CCL mit Filterung besser ab als MT. Allerdings nur um 3,82% (Fehlerrate MT: 58,50%; Fehlerrate CCL mit Filterung: 54,68%). Zu beachten ist, dass in Fig. 8 die Skalen auf den y-Achsen nicht übereinstimmen. Es sieht auf den ersten Blick so aus, als würden die Verfahren bei unterschiedlicher Anzahl an verrauschten Daten ungefähr die gleichen Ergebnisse liefern. Ein Blick auf die y-Achsen der Graphen zeigt aber, dass die Ergebnisse in unterschiedlichen Skalen gemessen wurden.
Wieso können die Methoden, die für das teilüberwachte Lernen eingesetzt werden, besser mit verrauschten Daten umgehen als der reine überwachte Ansatz? Wenn das Label eines Datenpunkts $x$, der zentral in der Klasse $c$ liegt, verändert wird, sodass dieser Datenpunkt das Label $c'$ hat, so wird das Lehrer-Netz diesen mit einer gewissen Wahrscheinlichkeit trotzdem der Klasse $c$ zuordnen. Das Lehrer-Netz folgt damit der *smoothness assumption*. Folglich wird das Schüler-Netz mit dem Datenpunkt $x$ und dem Label $c$ trainiert. Selbstverständlich wird das Lehrer-Netz nicht für alle falsch gelabelten Daten die richtige Klasse finden (z.B. wenn der Datenpunkt an dem Rand der Klasse liegt), aber es wird trotzdem für einen Teil der verfälschten Datenpunkte die richtigen Labels finden und somit besser performen als der überwachte Ansatz.

## 6 VERBESSERUNGEN DES ANSATZES

Als Erweiterung zu dem vorgestellten Ansätzen veröffentlichten die Autoren eine Fortsetzung des bereits präsentierten Papers. In dieser Forsetzung schlagen sie zwei Verbesserungen ihres Ansatzes vor. Als erstes eine Kombination der beiden Ansätze CCL mit Filterung und CCL mit Temperatur. Und als zweites eine Architektur mit mehreren Lehrer- und Schüler-Netzen. Beide Verfahren werden im Folgenden präsentiert und ihre Leistungsfähigkeit mit den bereits vorgestellten Verfahren verglichen.

### 6.1 Kombination von Filterung und Temperatur

Die Grundidee der Kombination von Filterung und Temperatur ist relativ einfach. Wenn sich das Lehrer-Netz mit einer Klassifikation relativ unsicher ist, dann soll das Schüler-Netz nicht nur abgemildert, sondern gar nicht trainiert werden. Wie beim Ansatz mit der Filterung wird zunächst eine nach der Unsicherheit der Klassifikation geordnete Liste der Datenpunkte aus dem Batch $B'$ gebildet. Im Anschluss daran werden wieder die $k$ sichersten Klassifikationen ausgewählt, mit denen das Schüler-Netz trainiert werden soll. Um erneut das Überanpassen an die als am sichersten eingeschätzen Datenpunkte zu verhindern, wird auch bei der Kombination von Filterung und Temperatur an dieser Stelle der probabilistische Filter angewendet, der zufällig eine definierte Anzahl an Datenpunkten aus den $k$ vielen Daten entfernt. Im Anschluss daran wird das Schüler-Netz mit den verbliebenen Datenpunkten trainiert. Dabei wird allerdings die Temperatur der einzelnen Datenenpunkte berücksichtigt, wodurch das Schüler-Netz mehr Wissen aus den sichereren Datenpunkten zieht, als aus den unsichereren.
Zu Beginn des Trainings, wenn $k$ kleiner ist, dominiert die Filterung den Trainingsprozess. Es werden nur die Datenpunkte, deren Klassifikation am sichersten ist, zum Trainieren des Schüler-Netzes genutzt. Im Laufe des Trainings wird allerdings das Trainieren mit der Temperatur immer bedeutungsvoller. Wenn sich $k$ immer weiter erhöht, dann werden

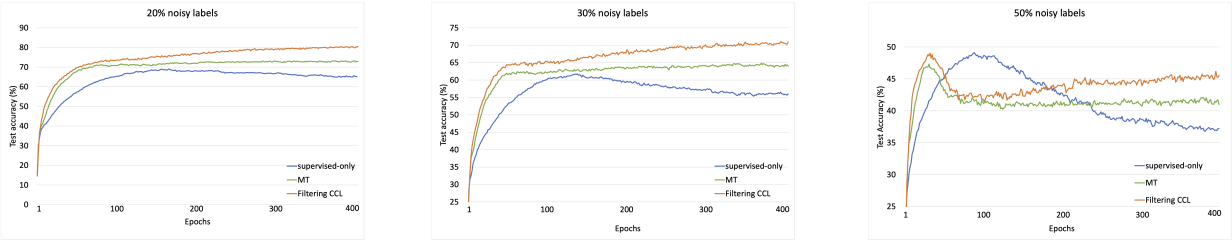


**Figure 8: Testgenauigkeit auf dem CIFAR-10 Datensatz mit 4.000 gelabelten Daten. Es wurden zufällig 20%, 30% und 50% der Labels verändert. Im supervised-Ansatz lässt sich sehr deutlich die Überanpassung an die falschen Labels ablesen. MT [15] und der vorgestellte Ansatz CCL mit Filterung zeigen eine gewisse Robustheit gegenüber den verrauschten Daten. Am besten schneidet der CCL-Ansatz mit Filterung ab.**

auch immer mehr Datenpunkte aus den Trainingsdaten zum Trainieren des Schüler-Netzes verfügbar. Das Verfahren wird dem CCL mit Temperatur immer ähnlicher.
Ob der neue Ansatz eine Verbesserung bringt und wie groß diese ausfällt wird in Abschnitt 6.3 untersucht. Zunächst wird aber noch eine zweite Verbesserung des bisherigen Ansatzes vorgestellt.

## 6.2 Mehrere Lehrer

Im bisherigen Ansatz waren das Lehrer- und das Schüler-Netz eng miteinander verbunden. Das Lehrer-Netz hat direkt bestimmt, mit welchen Datenpunkten das Schüler-Netz trainiert wird. Und andersherum hat sich das Lehrer-Netz durch das EMA immer mehr dem Schüler angenähert. Diese enge Verbindung kann die Kapazität des Models stark einschränken [9]. Um diese starke Verbindung zu lösen, haben die Autoren mehrere Paare aus Lehrer- und Schüler-Netzen eingeführt. Diese werden wie in Fig. 9 miteinander verbunden. Während der $teacher_i$ genutzt wird um den $student_{i+1}$ mit Hilfe des *certainty-driven consistency* Loss zu trainieren, werden die Gewichte von $teacher_i$ mit Hilfe von EMA durch den $student_i$ angepasst. Je nach Größe des entstandenen Kreises braucht eine, in einem Netz gewonnene Information, nun deutlich mehr Zeit, damit sie allen Netzen bekannt wird. Dadurch können alle Lehrer- und Schüler-Netze unabhängiger voneinander Lernen, was zu einer signifikanten Vergößerung der Kapazität des Modells führt. Ein weiterer Aspekt, den die Einführung weiterer Netze mit sich bringt, ist der Vorteil, dass eine größere Wahrscheinlichkeit besteht, dass die Netze zufällig gute Entdeckungen machen. Während im Standardmodell nur zwei Netze mehr oder weniger zufällig trainiert werden, werden nun mindestens 4 Netze zufällig trainiert. Die Wahrscheinlichkeit, dass dabei nutzbare Eigenschaften des Datensatzes gefunden werden, ist deutlich höher als bei der Standardvariante. Ein Beispiel dafür liefert bereits die Initialisierung der Netze. Durch die zufällige Initialisierung der Netze werden diese gerade zu Beginn des Trainings unterschiedliche Vorhersagen treffen. Da in der neu vorgestellten Architektur mehr Netze existieren und sich die guten Eigenschaften deren Initialisierung im Laufe des Trainings durch den Kreislauf ausbreiten sollten, könnte das Verfahren mit mehreren Lehrern eine Verbesserung darstellen.
Der nächste Abschnitt präsentiert eine Evaluation der beiden neu vorgestellten Techniken mit den bisher bekannten Ansätzen.

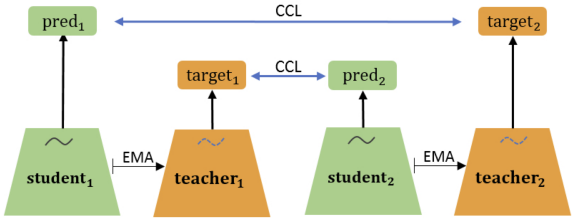


**Figure 9: Aufbau des Ansatzes mit mehreren Lehrer- und Schüler-Netzen. Der $teacher_i$ trainiert den $student_{i+1}$ und seine Gewichte werden mit Hilfe von EMA durch den $student_i$ angepasst.**

## 6.3 Evaluation

Die Durchführung der Experimente zur Bestimmung der Fehlklassifikationsrate ist mit dem Vorgehen in Abschnitt 5 vergleichbar. Im Gegensatz zur Originalarbeit geben die Autoren hier auch alle notwendigen Hyperparameter, wie z.B. die Lernrate (0,1), die Größe von $\alpha$ (0,99) oder den genutzten Optimierer (SGD) an. Des Weiteren wird erläutert, dass die Batchen $B'$, mit denen die Schüler-Netze trainiert werden, eine Größe von 512 haben, wovon 128 Datenpunkte gelabelt

| Model | CIFAR-10 | | | CIFAR-100 | SVHN |
|---|---|---|---|---|---|
| | 1000 labels | 2000 labels | 4000 labels | 10000 labels | 1000 labels |
| Filtering CCL | $14.35 \pm 0.51$ | $11.76 \pm 0.34$ | $9.77 \pm 0.16$ | $34.07 \pm 0.47$ | $3.70 \pm 0.23$ |
| Temperature CCL | $14.48 \pm 0.59$ | $11.84 \pm 0.33$ | $9.90 \pm 0.21$ | $34.19 \pm 0.42$ | $3.73 \pm 0.16$ |
| FT-CCL | $14.14 \pm 0.46$ | $11.03 \pm 0.24$ | $9.65 \pm 0.17$ | $33.92 \pm 0.36$ | $3.67 \pm 0.13$ |
| FT-CCL with 2 T | $13.68 \pm 0.38$ | $10.59 \pm 0.23$ | $9.17 \pm 0.19$ | $33.51 \pm 0.31$ | $3.53 \pm 0.18$ |
| FT-CCL with 3 T | $\mathbf{13.45 \pm 0.35}$ | $\mathbf{10.32 \pm 0.21}$ | $\mathbf{8.89 \pm 0.15}$ | $33.34 \pm 0.32$ | $\mathbf{3.50 \pm 0.17}$ |

**Figure 10: Evaluation der neuen Ansätze im Vergleich zum CCL mit Filterung und CCL mit Temperatur. Die Syntax ist aus Fig. 5 übernommen. Allerdings wurde hier nur über 5 Durchläufe gemittelt. Die Unterschiede im CCL mit Filterung und CCL mit Temperatur kommen wahrscheinlich durch die Wahl anderer Hyperparameter zu stande. Die jeweils besten gemessenen Ergebnisse sind hervorgehoben.**

und 384 Datenpunkte ungelabelt sind. Es ist möglich, wenn auch nicht abschließend zu klären, dass die Hyperparameter, mit denen die neuen Verfahren trainiert wurden, auch für die bereits bekannten Methoden eingesetzt wurden. Dies würde die Unterschiede in der Fehlklassifikationsrate des CCL mit Filterung und des CCL mit Temperatur erklären.
Fig. 10 zeigt, dass die Änderungen, die von den Autoren vorgenommen wurden scheinbar funktionieren. Bereits die Kombination von Filterung und Temperatur (FT-CCL) sorgt dafür, dass die Fehlklassifikationsrate weiter sinkt. Eine weitere Verbesserung lässt sich beobachten, wenn der zweite Verbesserungsvorschlag integriert wird. Bereits die Architektur mit 2 Lehrer- und Schüler-Netzen sorgt dafür, dass die Fehlklassifikationsrate weiter sinkt. Die für alle Aufgaben besten Ergebnisse liefert allerdings der Algorithmus, der mit jeweils drei Lehrer- und Schüler-Netzen arbeitet. Obwohl die Fehlklassifikationsrate von 33, 34% von keinem in Fig. 5 vorgestellten Verfahren für den CIFAR-1000 Datensatz unterboten werden konnte, ist dieses Ergebnis nicht markiert. Dies könnte daran liegen, dass auch die anderen aufgeführten Methode weiterentwickelt wurden und somit auch diese bessere Ergebnisse erzielt haben.

## REFERENCES


[1] O. Chapelle, B. Scholkopf, and A. Zien, Eds. 2009. Semi-Supervised Learning (Chapelle, O. et al., Eds.; 2006) [Book reviews]. *IEEE Transactions on Neural Networks* 20, 3 (2009), 542–542. https://doi.org/10.1109/TNN.2009.2015974

[2] Yanbei Chen, Xiatian Zhu, and Shaogang Gong. 2018. Semi-supervised Deep Learning with Memory. In *Computer Vision – ECCV 2018*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer International Publishing, Cham, 275–291.

[3] Cloudfactory. 2022. The Ultimate Guide to Data Labeling for Machine Learning. https://www.cloudfactory.com/data-labeling-guide. Accessed: 2022-07-25.

[4] Yarin Gal. 2016. *Uncertainty in Deep Learning*. Ph. D. Dissertation. University of Cambridge.

[5] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. https://doi.org/10.48550/ARXIV.1801.01290

[6] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. https://doi.org/10.48550/ARXIV.1503.02531

[7] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. 2009. The CIFAR dataset. https://www.cs.toronto.edu/%7Ekriz/cifar.html. Accessed: 2022-07-27.

[8] Samuli Laine and Timo Aila. 2016. Temporal Ensembling for Semi-Supervised Learning. https://doi.org/10.48550/ARXIV.1610.02242

[9] Lu Liu and Robby T. Tan. 2019. Certainty Driven Consistency Loss on Multi-Teacher Networks for Semi-Supervised Learning. https://doi.org/10.48550/ARXIV.1901.05657

[10] Yucen Luo, Jun Zhu, Mengxi Li, Yong Ren, and Bo Zhang. 2018. Smooth Neighbors on Teacher Graphs for Semi-Supervised Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8896–8905. https://doi.org/10.1109/CVPR.2018.00927

[11] Medium. 2022. Two Minutes of Semi-Supervised Learning. https://medium.com/dataseries/two-minutes-of-semi-supervised-learning-f0eb62729530. Accessed: 2022-07-25.

[12] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2017. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. https://doi.org/10.48550/ARXIV.1704.03976

[13] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. 2011. The Street View House Numbers (SVHN) Dataset. http://ufldl.stanford.edu/housenumbers. Accessed: 2022-07-27.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15, 56 (2014), 1929–1958. http://jmlr.org/papers/v15/srivastava14a.html

[15] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. https://doi.org/10.48550/ARXIV.1703.01780