DADA 2.0

28 December 2016

Fabian Lüthard

UZH, FS16
100 Jahre Dada
6 ECTS Kulturanalyse
Prof. Dr. Sandro Zanetti

Fabian Lüthard
Bremgartnerstr. 80
8003 Zürich
fabian.luethard@uzh.ch

DADA 2.0

Kann man 100 Jahre nach Entstehung des Dada noch Dada machen? Und falls ja, welche Formen könnte Dada annehmen? In diesem Projekt, welches eine Mischung zwischen einem Versuch eines digitalen Lautgedicht-Generators und einer Reflexion über selbigen ist, habe ich mich damit befasst wie Hugo Ball und co. ihre Lautgedichte geschrieben hätten, würden sie im digitalen Zeitalter leben. Der Reiz in dieser Aufgabe besteht mitunter darin, mit einer formalen Sprache mit sehr strengen Regeln was die Semantik und Syntax angeht, etwas zu schaffen, das diesen Regeln nicht folgen muss, und des weiteren auch keinen Regeln anderer Sprachen folgen muss. Es sind bloss Laute. In diesem Fall werden sie nicht einmal vom Menschen vermittelt, sondern rein maschinell verarbeitet. Das Projekt an sich ist nicht ganz frei von Widersprüchen, denn gewissermassen beisst sich ein Dada Lautgedicht-Generator auch mit dem Manifest Tristan Tzaras, wenn er sagt, dass sie, die Dadaisten genug von den kubistischen und futuristischen Akademien haben, weil es Laboratorien für formale Gedanken sind (38). Aber vielleicht ist es gerade wegen diesem Widerspruch genau richtig, das Projekt so anzugehen.

Mit einem einfachen Skript, welches nur minimalen formalen Anforderungen genügen muss, lassen sich innert einer Sekunde mehrere Zehntausend Laute oder Wörter generieren. Somit ist es nicht nur wie Richard Huelsenbeck in seinem Manifest geschrieben hat ein Nichts, sondern in gewisser Weise eine Überwindung der Sprache oder Sogar des Menschen selbst (33).

Bevor ich zu einigen Beispielen des Outputs des Generators sowie einer kurzen Reflexion über den digitalen Dada gelange, möchte ich kurz die Vorgehensweise sowie den Code selbst erläutern. Geschrieben wurde der Generator in der serverseitigen Skriptsprache PHP und kann daher ganz einfach in jedem Browser aufgerufen werden. Ich habe mehrere Versionen des Skripts gemacht, die mit verschiedenen Variablen und Funktionen spielen, und somit auch leicht andere Outputs geben. Ich werde den Code jedoch an der V3 erläutern. Nachstehend ist somit ein Bild aus dem Code Editor, da so noch die Syntax hervorgehoben wird.

dadagenv4.php | dadagenv3.php | dadamanifest2.tx | dadagenv5.php | Telem

```php
<?php

echo "WElCOME TO DADAGENERATOR V3";
echo "<br/>";
echo "<br/>";
//reads Hugo Ball manifest into a single string
$dadafile = file_get_contents("dadamanifest2.txt");

//read string into array
$clusterlength = rand(2, 6);
$dadarr = str_split($dadafile, $clusterlength);

//randomize array

shuffle($dadarr);

// count number of values in array to determine $randint si
$dadasize = count($dadarr);

//echo array

$i = 0;
foreach ($dadarr as $dadas) {
  $randint = rand(0, $dadasize);
  echo $dadarr[$randint] . "<br/>";
  echo $dadarr[$randint];
  echo $dadarr[$randint];
  echo $dadarr[$randint];
  if (++$i == 50) break;
}


?>
```

Als erster Schritt wird Quellmaterial gebraucht, mit welchem der Generator die Laute erzeugen kann. Eine Herangehensweise wäre gewesen, das gesamte Alphabet in ein sogenanntes „Array“ zu laden, oder idealerweise in zwei Arrays die nach Vokalen und Konsonanten getrennt sind, und dann eine Funktion zu schreiben, die diese Zeichen zu mehr oder minder sinnvollen Silben kombiniert. Ich habe mich jedoch für eine andere Herangehensweise entschieden, da ich eine direkte Verknüpfung von Dada zum Output wollte. Daher ladet der Generator nun zuerst ein Textfile von Hugo Balls Manifest, welches in die Variable „$dadafile“ geladen wird. Einfachheitshalber wurde der Text schon von Punktuation und überflüssiger Formatierung bereinigt, sodass der Generator nur noch auf Buchstaben zugreifen muss. Als nächstes wird zuerst definiert wie gross ein Cluster sein soll. In diesem Sinn ist ein Cluster ein Laut/Silbe. Hier wird mit „$clusterlength = rand(2, 6);“ die Variable so definiert, dass ein Cluster eine zufällige Länge zwischen zwei und sechs hat. Mit „$dadarr = str_split($dadafile, $clusterlength)“ wird ein Array names „$dadarr“ erzeugt, welches aus dem Manifest-Textfile den Text in Clusters zerstückelt, in der Länge, welche per Zufall von der Variable „$clusterlength“ angegeben wird. Mit „shuffle($dadarr);“ wird die Reihenfolge der nun zerstückelten Clusters im Array zufällig durchmischt. Mit „$dadasize = count($dadarr);“ werden die Elemente im Array $dadarr gezählt, da je nach ausgegebener Clusterlänge, die Anzahl an Elementen im Array anders sein kann.

Als letzter Schritt kommt die endgültige Ausgabe des Lautgedichts, beziehungsweise nicht nur die Ausgabe sondern auch der letzte Schritt der Randomisierung. Im Fall vom dadagenV3 geschieht dies mit einer sogenannten „foreach loop“. Zuerst wird eine Variable „$i = 0;“ festgelegt, welche als Zähler für die Iterationen (in diesem Fall entspricht das den Zeilen) festgelegt wird. Mit jeder Ausführung der „foreach loop“ erhöht sich der Zähler um eins. Die letzte Zeile der Loop legt somit auch fest, nach wie vielen Iterationen der Prozess abgebrochen werden soll. In diesem Fall habe ich die Anzahl auf 50 beschränkt, was ein Lautgedicht von 50 Zeilen ergibt. Ohne diese Beschränkung würde die Loop soviele Male laufen wie es die Variable „$dadasize“ vorschreibt, was wiederum je nachdem wie viele Elemente „$dadarr“ erhält mehr oder weniger lang sein könnte. Als letzten Schritt, sehe ich mich gezwungen, die „foreach“ Loop noch ein wenig genauer zu erläutern. Die erste Zeile leitet die Loop ein, und quantifiziert die Elemente im Array $dadarr als nutzbare Elemente „$dadas“, was schlicht und einfach eine formale Anforderung in PHP ist, welche die Elemente des Arrays für diese Loop klassifiziert und

verfügbar macht, aber hier weiter nicht interessant ist. Als nächstes wird mit „$randint = rand(0, $dadasize);“ die Variable festgelegt, welche zufälligerweise eine Zahl auswählt zwischen dem Element 0 (wo die Zählung im Array anfängt) und der vorher festgelegten Zahl, welche aus „$dadasize“ festgelegt wurde, also dem letzten Element des Arrays. In den darauffolgenden vier „echo“ Befehlen wird aus „$dadarr“ dieses zufällig ausgewählte Element wiedergegeben und im Browser angezeigt. In der letzten Zeile des „foreach“ Befehls wird die Bedingung angegeben, dass wenn 50 Iterationen durchgeführt worden sind, die Loop beendet wird, da sie ansonsten für jedes Element in „$dadarr“ ausgeführt wird, beziehungsweise soviel Mal, wie die numerische Grösse von „$dadasize“ ist. Wie der Output dieser Operationen ist, wird der Leser bald sehen. Zuerst jedoch eine Entschuldigung.

Leider, durch die bösartige Zerstückelung des Textes durch die Maschine, entstehen bei einigen Auslösungen des Generators zwischendurch auch Zeichenketten, welche dem dadaistischen Lautgedicht nicht unbedingt nahestehen. Ausserdem kann es auch sein, dass durch falsche Codierung auch teilweise Zeichen entstehen, welche nicht richtig angezeigt werden.

Bevor ich fortfahre, würde ich hier gerne einige Beispiele von zwei verschiedenen Versionen des Dada-Generators vorstellen. Hier, an erster Stelle, ist ein Beispiel der oben erläuterten Version V3:[1]

berüh

berühberühberühm Irrs

m Irrsm Irrsm Irrsavon r

avon ravon ravon redeite

edeiteedeiteedeite�rich

�rich �rich �rich sgang

sgang sgang sgang hologi

hologihologihologiit m�

it m�it m�it m�avon r

avon ravon ravon rn Dada

[1] luethard.com/dada/dadagenv3.php

n Dadan Dadan Dadaichter
ichterichterichterr ver
r verr verr ver der S
der S der S der Sorhabe
orhabeorhabeorhabea ist
a ist a ist a ist ist d
ist d ist d ist ddlem G
dlem Gdlem Gdlem Gtricht
trichttrichttricht Sprac
Sprac Sprac Spracdie No
die Nodie Nodie NoNur ei
Nur eiNur eiNur eiestus
estus estus estus e gewo
e gewoe gewoe geworan er
ran erran erran er von W
von W von W von Wung da
ung daung daung da Inde
Inde Inde Inded kein
d keind keind keinn Dada
n Dadan Dadan Dadada mhm
da mhmda mhmda mhmhologi
hologihologihologiDada T
Dada TDada TDada Ta ist
a ist a ist a ist dada
dada dada dadaacht
acht acht acht fügli
füglifüglifügliss er
ss er ss er ss er eben E
eben Eeben Eeben Efen ha
fen hafen hafen ha auf
auf auf auf uszuko

uszukouszukouszukor ver
r verr verr verbeck
beck beck beck all un
all unall unall unit m&#65533;
it m&#65533;it m&#65533;it m&#65533; es ge
es ge es ge es gesen wi
sen wisen wisen wiie Wel
ie Welie Welie WelGezier
GezierGezierGeziernden
nden nden nden

Und hier ein Beispiel der Version V5[2]:

oralis
oralisoralisGoethe
GoetheGoetheMachen
MachenMachensem Or
sem Orsem Oras Wor
as Woras Wores Her
es Heres Her Im D
Im D Im Dtun I
tun Itun Ite gan
te gante gankennen
kennenkennennden
nden nden &#65533;glich
&#65533;glich&#65533;glichtauche
tauchetauche Johan

[2] luethard.com/dada/dadagenv5.php

Johan Johan Schwe
Schwe Schwe es ge
es ge es geie Spr
ie Sprie Spr wo es
wo es wo esrte d
rte drte düberh
überhüberhg komm
g kommg kommden R&#65533;
den R&#65533;den R&#65533;is zur
is zuris zuras Wor
as Woras Worache z
ache zache zwo es
wo es wo es Das W
Das W Das Waupt e
aupt eaupt eWeltkr
WeltkrWeltkrr imme
r immer immean die
an diean die Dada
Dada Dada überh
überhüberh von W
von W von W sind
sind sind t selb
t selbt selbird D
ird Dird Dem Ans
em Ansem Anshätze
hätzehätzeie wir
ie wirie wirichtun
ichtunichtunLilien
LilienLiliene erfu
e erfue erfuran h&#65533;

ran hran hentspr

entsprentsprDada H

Dada HDada Hill Ko

ill Koill Koten da

ten daten dangt Je

ngt Jengt Jer mit

r mit r mit

Nun, da ich den Code und dessen Funktionsweise erläutert habe, und einige Beispiele gemacht sind, würde ich dazu noch gerne eine Reflexion anstellen, damit der Zusammenhang dieses Projekts mit Dada ein wenig klargestellt werden kann. Da der Lautgedicht-Generator als Basis Balls Manifest nimmt, werde ich mich hauptsächlich auf die Punkte, die er darin vorbringt, konzentrieren.

Wie schon angedeutet wurde, entsprechen diese generierten „Gedichte“ vielleicht nicht vollständig dem Geiste Dadas, da es sich um fast rein zufällige Zeichenketten handelt. Und doch ist diese Zufälligkeit etwas, was die Laute der Syntax und Semantik der Sprache enthebt, denn wie Hugo Ball schreibt, liest er am ersten Dada-Abend im Cabaret Voltaire „…Verse die nichts weniger vorhaben als: auf die Sprache zu verzichten“ (34) und dass man daran sehen könne, „wie die artikulierte Sprache entsteht. Ich lasse die Laute ganz einfach fallen“ (34). In diesem Sinn lässt auch der Generator die Laute fallen, jedoch nicht nach dem Lautgefühl des Menschen, sondern so wie die Maschine formalsprachlich angewiesen wurde die Laute fallen zu lassen. Die Randomisierung der Laute, wie sie der Dada-Generator macht, sollte jedoch ganz im Geiste Hugo Balls *Eröffnungs-Manifest* sein, denn durch diese Randomisierung wird auf die Sprache insofern verzichtet, als dass sie zwischen dem Eingangstext und der Ausgabe des Generators nicht-sprachliche Prozesse durchläuft, im Kern wird zwischen dem eingespeisten Text und dem später ausgegebenen Lautgedicht alles Zahl—Dada gefangen in den Schaltkreisen des Computers. Somit werden linguistische Zeichen durch den Einsatz via dem Binärweg von nonlinguistischen Zeichen generiert. Die Dada-Aufführung wird nicht mehr durch den Sprachapparat nach Lust und Laune des Vortragenden ausgeführt, sondern auf Befehl irgend eines Menschen auf der Welt, welchen den Dada-Generator aufruft. Es ist sozusagen ein „Dada

on Demand.“ Was Hierbei an sich noch interessant ist, ist dass die Struktur des Codes vom formalen Anspruch her eher an klassische Gedichte, als an freie dadaistische Lautgedichte erinnert. Genau wie ein Petrarca-Sonnett genau aus zwei Quartetten und zwei Terzetten mit dem immer-gleichen Schema bestehen muss, darf auch beim Code kein einziges formales Element von den Regeln abweichen. Nur schon ein fehlender Buchstabe, oder noch schlimmer, ein fehlendes Semikolon, führt zu einer Fehlermeldung und zum Scheitern des ganzen Projekts.

Wie man, aus den Beispielen des Generators, die ich an früherer Stelle aufgeführt habe sieht, haben sich aufgrund von Codierungsfehlern auch nicht-linguistische Zeichen eingeschlichen, was zuerst wie ein Störfaktor scheint, der beheben werden muss. Jedoch, wenn man es sich genauer überlegt, passen sie eigentlich ganz gut in das Thema, da sie für einen radikaleren Sprachverzicht sprechen, oder viel mehr für das unaussprechbare der Maschine, und fungieren so, in den Worten Wilkes „as a radical rejection of poetic tradition, with its failure to „write the word itself,“ and as the announcement of a counter-project involving a new kind of poetry that would operate outside the limits of conventional language“ (642). Dabei muss festgehalten werden, dass Wilke dabei die klassische Rezeption aufgreift, von der er später im seinem Text abweicht, beziehungsweise eine andere Dimension gibt. Um das mit dem Lautgedicht-Generator zu Verknüpfen, haben die Computergenerierten Lautgedichte anders als zum Beispiel Balls „gadji beri bimba“, die wie Wilke sagt, Gruppen von Elementen enthalten, welche in diversen natürlichen Sprachen vorkommen (660), nicht denselben Anspruch. Mit dem heutigen Stand der Technologie wäre es mit besseren Ressourcen durchaus machbar, einen Lautgedicht-Generator zu bauen, welcher mit natürlichen Elementen arbeitet, und diese dann statt nur in geschriebener Form ausgibt, diese auch gleich noch vorträgt. Wenn man jedoch von weniger hohen Ansprüchen ausgeht, und bedenkt, dass die Grundlage—gewissermassen die lautliche Grundlage—des Generators einer der Urtexte Dadas ist, und somit dies die natürliche Ur-Welt des Generators ausmacht, fungieren die rekonfigurierten Laute des Generators auch als Urlaute—als Urlaute des Grundstein Dadas an und für sich. Somit sollte eine digitale Transformation Dadas hundert Jahre nach seiner Entstehung legitimiert oder zumindest verziehen sein. Auch Projekte wie die digitalisierung des Dada-Archivs des Kunsthauses Zürich[3]

[3] www.kunsthaus.ch/dadadig/de/dada-digital-the-project

oder Dada-Data[4] zeigen, dass doch ein Wunsch besteht, Dada auch im digitalen Zeitalter zu erhalten. Als intermediale Bewegung ergibt es auch durchaus Sinn, Dada in die digitale Welt zu transportieren, insbesondere wenn man Kittler in seiner aussage glaubt, dass im optoelektrischen Kanal alle Medien zusammenlaufen (7). Somit kann der optoelektrische Kanal im engen Sinne noch immer nicht die Bombe übertragen, aber zumindest kann er uns das dadaistische Gedankengut wohl noch mindestens bis zum nächsten Jubiläum erhalten.

Wenn auch die Lautgedichte des Generators zeitweise nicht zufriedenstellend sind, so könnte man wohl den Quelltext des Generators, mit den Dada-thematischen Variablen doch, so behaupte ich, im weitesten Sinn gesehen als dadaistisches Gedicht vortragen. Somit hätte Dada auch noch mindestens eine formale Sprache entfremdet. Und somit hätte man, ganz im Sinne Dadas eine neue Kunstrichtung. Und somit möchte ich damit schliessen, indem ich die Worte Hugo Balls entfremde. Wir müssen nicht überall unseren Mund dran hängen, denn: Der Code, der Code, das Weh gerade an diesem Ort und in dieser Zeit, meine Damen und Herren, ist eine öffentliche Angelegenheit ersten Ranges.

[4] www.dada-data.net/de/

Zitierte Werke

Ball, Hugo. (1994). “Eroeffnungs-Manifest, 1. Dada-Abend.” In: *DADA total: Manifeste, Aktionen, Texte, Bilder*. (pp. 34). Stuttgart: Reclam.

Huelsenbeck, Richard. “Erklärung.” In: *DADA total: Manifeste, Aktionen, Texte, Bilder* (pp. 33). Stuttgart: Reclam.

Kittler, Friedrich A. (1986). *Grammophon, Film, Typewriter*. Berlin: Brinkman & Bose.

Tristan Tzara. (1994). “Manifest Dada 1918*.” In: *DADA total: Manifeste, Aktionen, Texte, Bilder*. (pp. 38). Stuttgart: Reclam

Wilke, Tobias. (1994). “Da-da: ‘Articulatory Gestures’ and the Emergence of Sound Poetry. *MLN* 128:3. (pp. 639-668). Baltimore: Johns Hopkins University Press.